FORMULAS AND ALGORITHMS FOR OPTIMIZING THE PERFORMANCE OF RAPIDLY CHANGING SATELLITE NETWORKS

By

CHARLES D. MCLOCHLIN

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1989

ACKNOWLEDGEMENTS

Dr. Chow and Dr. Newman-Wolf deserve thanks for directing my research and providing guidance in the writing of this dissertation. I would also like to thank the other committee members for their helpful comments.

Dr. Principe and Dr. Papachristidis expanded my way of thinking and without their help I would not have completed the requirements for the Ph. D. degree.

Dr. Christopher Ward provided the Latex macros for the formating of this dissertation.

Financial support for this research was supplied in part by an SDIO/IST contract administered by the Department of Navy, Space and Naval Warfare Systems Command under contract number N00039-87-C-0221.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

FORMULAS AND ALGORITHMS FOR OPTIMIZING THE PERFORMANCE OF RAPIDLY CHANGING SATELLITE NETWORKS

By

CHARLES MCLOCHLIN

May 1989

Chairman: Dr. Yuan-Chieh Chow
Cochairman: Dr. Richard Newman-Wolfe
Major Department: Computer and Information Sciences

Rapidly changing satellite networks are now in the design stage. Interest in this new area has been fostered by the Strategic Defense Initiative (SDI). Many SDI architectures have been proposed, but because the technology has not been fully developed and no satellite-to-satellite links exist, the architecture is only loosely defined.

This dissertation considers the SDI communications network from a systems point of view. Research was performed in the areas of topology, link assignment, routing, models, and performance measures. At the systems level, these areas are closely related. For example, consider the physical network which is a topology. In order to determine the performance of the system model using packet delay as a metric, a link assignment and routing algorithm are required.

The emphasis of this research was to develop link assignment and routing algorithms for optimizing the performance of rapidly changing satellite network topologies. Performance measures for the link assignment included connectivity, retargeting frequency, and propagation delay. Measures for routing include end-to-end delay,

rerouting frequency, and number of common satellites on multiple paths between origin and destination. Minimizing the number of common satellites on multiple paths is an important consideration for survivable SDI communications and is a new area of research. Efficient algorithms for this problem are presented.

The topology optimization of satellite networks used a model of Np (orbit planes) and Ns (satellites per orbit) which provided complete coverage of the earth at all times. Closed formulas for the Np,Ns model were derived for the minimum altitude, minimum propagation delay, and maximum propagation delay. Algorithms were developed to obtain minimum, average, and maximum propagation delay for arbitrary single altitude topologies.

Optimization of link assignment for arbitrary topologies uses a 2-level hierarchical model based on an Np,Ns region concept. The regions are circles which cover all points on an arbitrary altitude shell. Level 1 satellites form a mesh backbone between regions which contain level 2 satellites. This link assignment has an optimal connectivity when satellites are constrained to have four antennas. An analytical solution for the total delay of the 2-level hierarchical model is derived for region disjoint path routing.

The routing research extends the minimum cost flow problem to include a minimum number of common nodes on multiple paths. The general minimum cost flow problem finds paths of minimum cost without regard to repeated nodes, which is undesirable for survivable network routing.

CHAPTER 1
INTRODUCTION

1.1 Scope

A new kind of communication network is evolving which incorporates satellite-to-satellite links. Communication bandwidths of 10-100 megabits using full-duplex laser cross-links are envisioned. The links will be redirected in order to optimize performance or prevent network fragmentation. These networks will not use geosynchronous orbiting satellites, hence continuous communication between earth stations via a single satellite will not be possible. Interest in this new area has been fostered by the Strategic Defense Initiative (SDI). The SDI requirements for sensing satellites, weapon satellites, and battle manager satellites cannot be met by using only the geosynchronous altitude of 35,744 km [1,5,9]. Lower altitudes afford a higher resolution view of land-based enemy missiles, and higher altitudes offer a greater immunity to attack. Consequently, sensor and weapon satellites are more effective below the geosynchronous altitude and battle managers are more secure above.

Many SDI architectures have been proposed, but because the technology has not been fully developed and no satellite-to-satellite links exist, the architecture is only loosely defined. In addition, because SDI is a multi-service venture, the architecture can vary to reflect each service's respective battle phase responsibility [2]. The three battle phases with responsible service are:

1. Boost - enemy vehicle lift-off (Air Force)

2. Midcourse - warhead dispersement (Navy)

3. Terminal - atmosphere re-entry (Army)

The architecture has been evolving and will continue to evolve as research progress is made. The Harris Corporation has contributed to this research. They have been using architectures provided by the Rome Air Development Center (RADC) and Naval Research Laboratory (NRL). A summary of the published Harris work is provided in Chapter 2.

A primary objective of the SDI communications network is that it must be survivable. The network should be able to adapt to jammed links, satellite losses, and traffic pulses. In addition, the network should operate at near optimum performance measures. Obviously, these are high goals and because the architecture is not firmly established, many areas of research are available. This research proposal considers:

1. topology
2. link assignment
3. routing
4. models
5. performance measures

Although these areas are very diverse from a purely theoretical view, they are closely related from a systems view. For example, consider the physical network which is a topology. In order to determine the performance of the system model using packet delay as a metric, a link assignment and routing algorithm are required.

1.2 Principal Results

The study of low-altitude satellite network topologies has resulted in the derivation of several important closed formulas and algorithms. These equations and algorithms have been used to facilitate and reduce simulation of satellite networks. In addition, they have been used to optimize single coverage satellite topologies. References [26,27] provide simulation results and applications of the equations and algorithms. The derivations and proofs are in reference [28]. The principal equations derived include:

1. Maximum visibility distance of satellites with specified minimum horizon angle;
2. Minimum altitude of an Np (orbit planes), Ns (satellites per plane) satellite topology with specified minimum horizon angle;
3. Minimum propagation delay between arbitrary points on the earth via an $Np = 2$, $Ns \geq 3$ topology with specified minimum horizon angle;
4. Maximum propagation delay for maximally separated points on the earth via an $Np = [2,3,4,5]$, $Ns = [2,4,6,8,10,12]$ topology at minimum altitude with specified minimum horizon angle.

Two propagation delay algorithms are given with their pseudo-code and correctness proofs. The algorithms

1. determine the minimum propagation distance for arbitrary points on the earth separated by an arbitrary set of terrestrial arc distances, for a single altitude topology and specified minimum horizon angle; and

2. determine the time average and maximum propagation delay between a set of terrestrial source-destination pairs via a single altitude topology with specified minimum horizon angle.

The minimum propagation delay algorithm is very efficient because it uses closed formulas and dynamic programming rather than a grid search. The time average and maximum propagation delay algorithm uses dynamic programming to provide a speed-up of 10 over Dijksta's shortest path algorithm for a topology of 45 satellites at 1000 km.

An Np,Ns region model was developed to evaluate the performance of arbitrary 2-altitude topologies. A closed form for the minimum altitude of the Np,Ns regions is derived. Also an analytic solution for the total delay of a 2-level hierarchical satellite network is presented. References [10,25] contain applications of the model which include an optimal link assignment and survivable routing strategy for a region disjoint structure.

Several multiple path routing algorithms have been developed which find K paths of minimum cost, where cost is defined as delay or remaining time of path. In addition, the K paths have a minimum number of common nodes and links. The running time of finding the K paths from origin to destination is $O(n^2)$, where n is the number of satellites and $K \ll n$.

1.3 Chapter Synopsis

The chapters are ordered starting with topology and closed formulas for propagation delay, then progress toward the multiple path algorithms and multiprocessor simulation for the SDI communications network. Chapter 2 provides background material on the features and requirements of the SDI architecture. Also, a brief review

of the Harris work and others on link assignment, routing, and simulation is presented. A general description of the minimum cost flow problem and path diversity is also given. Chapters 3 and 4 provide derivations of the closed forms and algorithms developed to reduce simulation time. These chapters also contain some simulation results. Chapter 5 presents a 2-level hierarchical model which is used to derive the total delay of a 2-altitude SDI architecture. Chapter 6 presents the K-paths algorithms of minimum total cost. Chapter 7 discusses the multiprocessor simulation for the SDI communications network. Finally, Chapter 8 contains the conclusions of this dissertation.

CHAPTER 2
BACKGROUND

A brief overview of the SDI architecture issues and the published Harris work on the communications network is provided in this chapter. Also a description of the minimum cost flow problem and node disjoint path algorithms are given. Section 2.1 gives the features and 2.2 discusses the requirements. The second generation Harris link assignment and routing strategy is given in section 2.3, and other relevant work is presented in 2.4. The minimum cost flow problem is stated in section 2.5, and 2.6 discusses algorithms for node disjoint paths of minimum total cost.

2.1 Features of the SDI Architecture

The two basic capabilities of SDI are to sense enemy vehicles (missiles) and to disable them. These capabilities are very diverse and, because of hardware specialization, separate satellite types called sensors and weapons have been proposed. In addition, a third type of satellite called a battle manager is used to process sensor reports and direct the weapons. Due to visibility resolution constraints, the sensor and weapon satellites are placed in low-altitude orbits. Atmospheric drag poses a lower limit of about 500 km.

There are many different solutions to the problem of assimilating sensor traffic and disabling the corresponding targets as efficiently as possible. Each solution can be used to define a different kind of architecture. NRL has defined an architecture which reflects a compromise between a completely distributed architecture and a centralized one. Approximately 12 battle manager satellites are used at an altitude of about

50,000 km. This high altitude offers a visibility probability between weapons and sensors of 0.5 and also security from terrestrial attack. A weakness of this approach is that enemy space mines need only disable 12 satellites to render SDI useless.

Cost reductions in laser links and sensors, along with advances in distributed processing, may eliminate the need for a battle manager satellite. Hence a network of sensors with high-speed processing capability would provide the battle manager function, and sensors would direct the weapon satellites. This architecture has the advantage of more graceful degradation with satellite loss. In addition, the communications delay between sensors and battle managers has been eliminated. Future Harris work will use larger numbers of satellites, and the satellites will be less distinctive (i.e., a blending of sensor, weapon, and battle manager functions).

The SDI network will use high-bandwidth, point-to-point communications between sensor satellites. Broadcast links are also included in some architectures such as the NRL plan, which uses time division multiplexing between the weapon and battle manager satellites. A broadcast channel has also been proposed for finding satellites which become disconnected from the network. However, because broadcasting requires more power and is more vulnerable to jamming, point-to-point communications are being substituted for broadcast links whenever practical.

2.2 Requirements

The two basic requirements for the SDI communications network are that it be survivable and near optimum. Unfortunately, neither of these requirements have unique definitions. Reference [32] gives survivability in terms of two measures: 1) the end-to-end communications delay as a function of failed nodes in the weapons network, and 2) the number of isolated nodes in the weapons network as a function

of the number of failed nodes. Reference [27] gives survivability in terms of connectivity and ground coverage. The reason for so many different definitions is that there are numerous performance measures for a communications network. Any of these measures can be optimized, but no all single encompassing architecture will simultaneously optimize all measures. A classic contradiction for satellite networks is the minimization of propagation delay (low altitude) and maximization of connectivity (high altitude). Hence, survivability means many things. For purposes of this dissertation, delay, connectivity, retargeting frequency, and number of common nodes on multiple paths will be the measures of survivability. The first two choices are the ones generally used for networks, the latter two being a specialization for SDI.

2.3 Summary of Harris Work

The link assignment algorithm performs three functions which include: (1) establishing and maintaining a single connected subnetwork, (2) connecting the subnetworks into a single network, (3) optimizing the connectivity within subnetworks [7]. The algorithm is run on all satellites. Each satellite has an identical copy of the network connectivity table. A heuristic is used with various metrics consisting of propagation delay, remaining link visibility, connectivity, queueing delay, etc., for determining the link assignment.

The routing algorithm uses multiple disjoint paths for sending multiple message copies [8]. The redundancy adds to the traffic flow, but offers greater survivability. The disjoint paths are computed by using Max Flow. A heuristic is used for load balancing which uses a metric of queueing delay and propagation delay.

2.4 Other Related Work

Reference [37] gives a link assignment algorithm and simulation data for ground coverage and propagation delay vs. retargeting time. The link assignment algorithm

uses the all pairs propagation delay as an objective function. A heuristic is used to minimize the propagation delay. The heuristic assigns links to adjacent neighbors on the same plane and then connects satellites on adjacent planes until the number of satellite antennas are exhausted. For four antennas and N_p, N_s topologies, the final link assignment is very similar to the N_p, N_s mesh link assignment described in Chapter 5.

Reference [11] uses random and deterministic routing on two satellite constellations to determine output statistics which include end-to-end delay and average queue sizes. Chapter 5 derives similar results for the N_p, N_s mesh link assignment.

Reference [20] gives a hierarchical routing algorithm which is based on (1) a hierarchical addressing scheme, (2) regional node routing architecture.

References [13.24] discuss simulation programs for satellite networks using a single processor. Satellites are modeled as either procedures or processes. Both programs were developed as tools to evaluate the performance of rapdily changing networks. The programs are event driven.

2.5 Minimum Cost Flow Problem

The desire to reduce transportation costs and risks led to the formulation of this problem in the late 1940s. Several routes and transportation media were available between warehouses and troops. However, the capacity and cost (risk) varied with the route. Several people from various countries worked independently on this problem during World War II. Two of the pioneers of this era later wrote a text [18] which has become a classic on the subject of flow. The text contains algorithms for solving various flow problems. Other researchers have improved the efficiency of the algorithms, notably [15] which uses a flow augmentation along a shortest path [14].

The minimum cost flow problem can be solved by linear programming as suggested in [12], but reference [29] describes this approach akin to killing a mouse with a cannon. However, many theorems on flow are proved using the principle of linear programming.

Some related work includes an algorithm for the minimum augmentation of a directed tree to a K-edge-connected directed graph [23].

2.6 Disjoint Paths of Minimum Total Cost

References [19,35] constrained the general minimum cost flow problem to a capacity of 1 and showed how finding node disjoint paths of minimum cost was as simple as finding a shortest path. However, a similiar problem, one of finding a maximum number of bounded paths, is shown to be NP-complete in [21,33].

Efficiency improvements in Dijkstra's shortest path algorithm for sparse graphs is given in [22] which uses a d-heap. This approach was later used in [36] for an $O(m \log n)$ algorithm for finding two, edge disjoint paths between a single source and n destinations on a graph of m edges.

Some related work to disjoint paths includes optimally reliable graphs [16] and efficient all-pairs shortest-path algorithms [30].

CHAPTER 3
CLOSED FORMS FOR PROPAGATION DELAY AND CONNECTIVITY

A simple model using Np orbit planes and Ns satellites per orbit is analyzed to obtain closed formulas for measuring performance. The following sections contain derivations of the performance metrics of propagation delay and connectivity. Section 3.1 gives the condition for satellite visibility with non-zero minimum horizon angle. This condition applies to any single altitude topology. The minimum altitude for complete coverage is produced in section 3.2. The minimum and maximum propagation delays are derived in sections 3.3 and 3.4 respectively. Connectivity is discussed in section 3.5, and a proof that three, time invariant node disjoint paths exist for the Np,Ns model is given. Finally, section 3.6 shows how the equations can be used for topology optimization. An abbreviated version of this section is in [27].

3.1 Maximum Visibility Distance

Definitions:

1. Visibility - direct line of sight exists between a point on the earth and a satellite or between two satellites
2. Minimum horizon angle - the minimum angle between a tangent to the earth and a satellite

Symbols:

1. R_e - the radius of the earth, approx. 6378 km
2. A - altitude

3. H_a - horizon angle

Lemma 3.1: The maximum tangential distance at which a point on the earth and a satellite have visibility at altitude A is

$$D_{max}/2 = A\sqrt{1+2\frac{R_e}{A}} \tag{3.1}$$

It follows that the maximum distance at which two satellites have visibility is D_{max}.

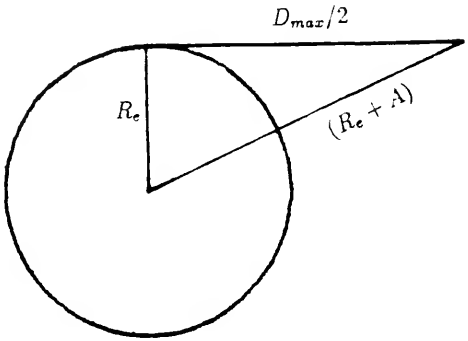


Figure 3.1. Maximum Tangential Visibility Distance

Proof: The triangle with vertices at the center of the earth, the center of the satellite, and the point of tangency on the earth is a right triangle. Orbital mechanics requires every satellite orbit to be in a plane which includes the center of the earth. By application of the Pythagorean theorem,

$$D_{max}/2 = A\sqrt{1+2\frac{R_e}{A}}$$

Lemma 3.2: The maximum distance at which a point on the earth and a satellite have visibility with minimum horizon angle H_a is

$$\begin{aligned} D'_{max}/2 &= \frac{\cos(\theta) D_{max}}{2\cos(\theta + H_a)} \text{ where} \\ \theta &\equiv \arctan\left(\frac{D_{max}}{2R_e}\right) \end{aligned} \tag{3.2}$$

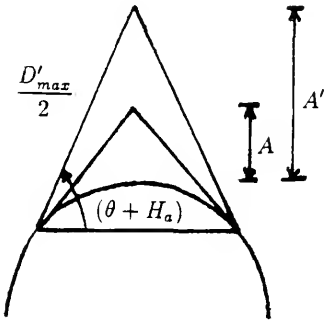


Figure 3.2. Maximum Visibility Distance with Minimum Horizon Angle

Proof: The equilateral triangles with satellite at altitude A and satellite at altitude A' have a common base. Using the cosine relations of angle θ and $\theta + H_a$ results in

$$\begin{aligned} D'_{max}/2 &= \frac{\cos(\theta) D_{max}}{2\cos(\theta + H_a)} \text{ where} \\ \theta &\equiv \arctan\left(\frac{D_{max}}{2R_e}\right) \end{aligned}$$

Lemma 3.3: For some altitude A with zero minimum horizon angle, the new altitude for a specified minimum horizon angle, H_a is given by:

$$A' = \frac{D'_{max}}{2}\sqrt{1-\cos^2(\theta+H_a)} - (1-\cos(\theta))R_e \tag{3.3}$$

Proof: From figure 3.2, h' and A' can be expressed as:

$$h' = \frac{D'_{max}}{2}\sqrt{1-\cos^2(\theta+H_a)}$$
$$A' = h' - (1-\cos(\theta))R_e$$

Hence the equation follows.

3.2 Minimum Altitude for the Np,Ns Topology

Definition of the Np,Ns Satellite Topology:

1. There are Ns equally spaced satellites per plane, with $Ns \geq 3$
2. There are Np equally spaced planes, with $Np \geq 2$
3. The $Ns \times Np$ satellites are at a single minimum altitude which provides complete coverage of a spherical earth
4. The phase offset of the satellite planes is zero
5. All planes are rotated about one axis

Lemma 3.4: The locus of points on the earth $D_{max}/2$ away from a satellite is a circle of radius R_c

$$R_c = R_e\frac{D_{max}}{2(R_e+A)} \tag{3.4}$$

Proof: Without loss of generality, assume the z-axis intersects the satellite. Using spherical coordinates, the distance between a point on the earth and the satellite is:

satellite location	earth location
$S_x = 0$	$E_x = R_e \sin(El)\cos(Az)$
$S_y = 0$	$E_y = R_e \sin(El)\sin(Az)$
$S_z = R_e + A$	$E_z = R_e \cos(El)$

$$D_{max}^2/4 = E_x^2 + E_y^2 + (E_z - S_z)^2$$

This equation simplifies to $\cos(El) = \frac{R_e}{R_e+A}$. Using the triangle in Lemma I. $\sin(El) = \frac{D_{max}}{2(Re+A)}$. Thus the circle

$$E_x^2 + E_y^2 = R_e^2 \sin^2(El)$$

has radius R_c

$$R_c = R_e \frac{D_{max}}{2(R_e + A)}$$

Lemma 3.5: For the Np,Ns topology with all orbital planes rotated about the x axis, the hardest points to cover (fewest satellites overhead) are in the y-z plane.

Proof: Without loss of generality, let two adjacent orbital planes be rotated by $\pm\frac{\pi}{2Np}$ radians about the x-axis. Using polar coordinates with Az measured from the x-axis in the x-y plane, a satellite location on the lower plane is given by

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\frac{\pi}{2Np}) & sin(\frac{\pi}{2Np}) \\ 0 & -\sin(\frac{\pi}{2Np}) & \cos(\frac{\pi}{2Np}) \end{pmatrix} \begin{pmatrix} (R_e + A)\cos(Az) \\ (R_e + A)\sin(Az) \\ 0 \end{pmatrix}$$

The distance between satellites on adjacent orbital planes is given by the absolute difference between their z coordinates which is $2(R_e + A)\sin(Az)\sin(\frac{\pi}{2Np})$. Hence the maximum distance occurs in the y-z plane at $Az = \frac{\pi}{2}$. By Lemma 3.4 the earth coverage provided by a satellite is a circle, so the hardest point to cover will be the intersection of arcs connecting centers of diagonal circles as shown in figure 3.3. The

distances between the centers of the circles (1,4) and (2,3) are at a maximum when the satellites are at $Az = \frac{\pi}{2} \pm \frac{\pi}{Ns}$ radians. This can be shown by taking the derivative of the distance equation for the vertical separation of satellites and noting the rates of change.

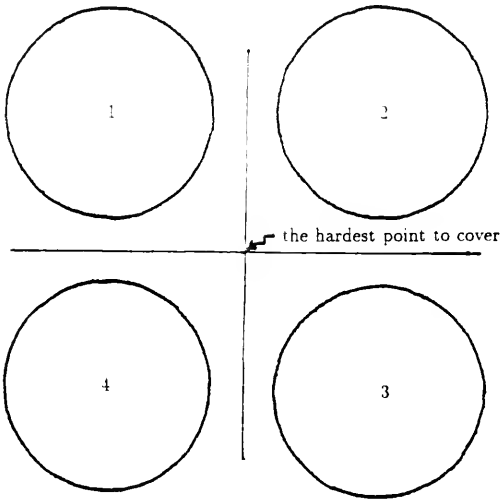


Figure 3.3. Satellites 1,2,3,4 Looking down the Y-Axis

$$
\begin{aligned}
Dz13 &= derivative\_of\_the\_distance\_between\_satellites(1,3) \\
&= 2(R_e + A)\sin(\frac{\pi}{2Np})\sin(Az') \\
Dz24 &= derivative\_of\_the\_distance\_between\_satellites(2,4) \\
&= 2(R_e + A)\sin(\frac{\pi}{2Np})\sin(Az' - \frac{2\pi}{Ns})
\end{aligned}
$$

where

$$
Az' = \frac{\pi}{2} - Az
$$

$$0 \leq Az' \leq \frac{\pi}{Ns}$$

The horizontal separation remains constant and the distance between satellites (2,4) decreases at a faster rate than (1,3) increases as Az' is varied from $\frac{\pi}{Ns}$ to zero. Hence the maximum diagonal satellite spacing and corresponding circles of coverage are furthest apart when $Az' = \frac{\pi}{Ns}$, and the intersection of the diagonals in the y-z plane is at the hardest point to cover.

Theorem 3.1: The minimum altitude for the $Np.Ns$ satellite topology is

$$A_{min} = \left(\frac{1}{\cos(\frac{\pi}{Ns})\cos(\frac{\pi}{2Np})} - 1 \right) R_e \tag{3.5}$$

Proof: By Lemma 3.5 the hardest point to cover is in the y-z plane and since the orbital planes are symmetrically rotated about the x axis, then it is sufficient to show that if the earth location intersected by the y axis is covered by a satellite at all times, then all points on the earth are covered at all times. The topology of Ns satellites per plane ensures that a satellite will always be within $\frac{\pi}{Ns}$ radians of the y-z plane. The equations for the satellite position in Lemma 3.5 can be used to compute the distance between the earth location ($E_x = 0, E_y = R_e, E_z = 0$) and a satellite in the azimuth range ($\frac{\pi}{2} - \frac{\pi}{Ns}$) radians to $\frac{\pi}{2}$ radians. Setting this distance to the visibility distance of $D_{max}/2$ ensures all points will be covered.

$$\begin{aligned} D_{max}^2/4 &= (E_x - S_x)^2 + (E_y - S_y)^2 + (E_z - S_z)^2 \\ &= R_e^2 + (R_e + A)^2 - 2R_e(R_e + A)\cos(\frac{\pi}{2Np})\sin(Az) \end{aligned}$$

The right side of the above equation is maximized over the range of Az when Az $= \frac{\pi}{2} - \frac{\pi}{Ns}$. Using this value for Az ensures that the hardest point to cover on the

earth will always be visible to the satellite over the range of Az. Solving for A in the above equation with $\mathrm{Az} = \frac{\pi}{2} - \frac{\pi}{Ns}$ produces the minimum altitude for complete coverage.

$$A_{min} = \left(\frac{1}{\cos(\frac{\pi}{Ns})\cos(\frac{\pi}{2Np})} - 1 \right) R_e$$

The value for A_{min} assumes a zero horizon angle. For some specified minimum horizon angle, H_a, Lemma 3.3 can be used to find the new altitude. The new altitude A'_{min} with D'_{max} evaluated at A_{min} is:

$$A'_{min} = \frac{D'_{max}}{2}\sqrt{1 - \cos^2(\theta + H_a)} - (1 - \cos(\theta))R_e \tag{3.6}$$

3.3 Minimum Propagation Delay for the Np,Ns Topology

Lemma 3.6: The shortest geodesic (minimum distance path) using three or more satellites on an arc of C_t radians occurs when the maximum possible satellite spacing is used between as many satellites as possible.

Proof: Let the shortest geodesic use n ($n \geq 3$) satellites on an arc of C_t radians and radius R with $\pi \geq C_t = C_1 + C_2 + .. + C_{n-1}$, where C_i is the arc between satellite i and satellite i+1. The path distance of the n satellites is

$$path\_distance = 2R(\sin(\frac{C_1}{2}) + \sin(\frac{C_2}{2}) + \cdots + \sin(\frac{C_{n-1}}{2}))$$

Since $\frac{\sin(x)}{x}$ is monotonically decreasing for $0 \leq x \leq \frac{\pi}{2}$, then

$$\begin{aligned} min\_path\_distance &= 2R(k\sin(\theta) + \sin(\frac{C_t}{2} - k\theta)) \text{ where} \\ k &= \lfloor \frac{C_t}{\theta} \rfloor \\ \theta &\equiv \arctan\left(\frac{D_{max}}{2R_e}\right) \end{aligned}$$

Hence the shortest geodesic uses the maximum possible satellite spacing between as many satellites as possible.

Lemma 3.7: The minimum propagation path will have equal source-to-satellite and destination-to-satellite distances; and source, destination, center of the earth, and both terminating satellites will be in one plane.

Proof: Let the terrestrial arc distance be s, with source and destination separation $\frac{s}{R_e}$ radians. Without loss of generality, the terminating satellites can be symmetrically located about the y-axis as shown in figure 3.4. The objective is to show e is $\frac{s}{2R_e}$ and $Y_{rot} = 0$ for a minimum length path. This proves Ds1 = Ds2 and points are coplanar.

source and destination location
$S_x = -R_e \sin(\frac{s}{R_e} - e)$
$S_y = R_e \cos(\frac{s}{R_e} - e)\cos(Y_{rot})$
$S_z = R_e \cos(\frac{s}{R_e} - e)\sin(Y_{rot})$
$D_x = R_e \sin(e)$
$D_y = R_e \cos(e)\cos(Y_{rot})$
$D_z = R_e \cos(e)\sin(Y_{rot})$

satellite 1,2 locations
$S_{x1} = (R_e + A)\sin(\phi)$
$S_{y1} = (R_e + A)\cos(\phi)$
$S_{z1} = 0$
$S_{x2} = -S_{x1}$
$S_{y2} = S_{y1}$
$S_{z2} = S_{z1}$
ϕ = arbitrary angle
$Ds1$ = source to satellite 1 distance
$Dd2$ = destination to satellite 2 distance
$D12$ = satellite 1 to satellite 2 distance
$path\_length = Ds1 + Dd2 + D12$

The derivative of the path with respect to e is zero when e is $\frac{s}{2R_e}$ for all values of Y_{rot} and ϕ. The path length is minimized for all ϕ when $Y_{rot} = 0$. Hence the source, destination, center of the earth, and both terminating satellites are in one plane. Also Ds1 = Ds2 since $e = \frac{s}{2R_e}$.

Theorem 3.2: The minimum propagation path over an arbitrary terrestrial arc distance s, between all pairs of points for the $Ns..Np = 2$ satellite topology is given by:

$$path\_distance = 2\sqrt{A^2 + 2R_e(R_e + A)(1 - cos(\frac{s}{2R_e}))} \text{ where} \qquad (3.7)$$
$$s = (0, 2R_e\theta)$$

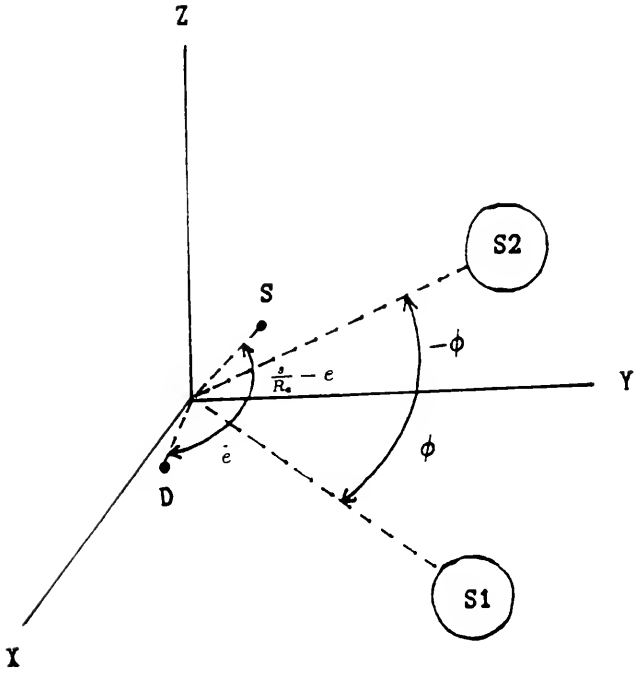


Figure 3.4. Minimum Path Distance Via Terminating Satellites

Proof: For $Np = 2$, $\theta \leq \frac{\pi}{2}$ and only one satellite is needed to connect all points on the earth. By Lemma 3.7, the source-to-satellite distance equals the destination-to-satellite distance. Hence the path distance equation follows for zero minimum horizon angle. For some specified minimum horizon angle H_a, A' can be substituted for A.

3.4 Maximum Propagation Delay for the $Np.Ns$ Topology

Theorem 3.3: The maximum geodesic propagation path over the terrestrial arc distance, s = $R_e\pi$, between all pairs of points for the $Ns \geq 4$ and $Np = [2,3,4,5]$ satellite topology with altitude A, and visibility distance D_{max} is given by:

$Np = 2, Ns = (4, 6, 8, 10, 12)$	$D2 = 2D_{max} + 2(R_e + A)\cos(\frac{pi}{Ns})\cos(\frac{\pi}{4})$
$Np = 3, Ns = (4)$	$D3 = 2D_{max} + 2(R_e + A)\cos(\frac{pi}{Ns})\sin(\frac{\pi}{3})$
$Np = 3, Ns = (6)$	$D3 = D_{max} + 6(R_e + A)\sin(\frac{pi}{Ns})$
$Np = 3, Ns = (8, 10, 12)$	$D3 = 2D_{max} + 4(R_e + A)\cos(\frac{pi}{Ns})\sin(\frac{\pi}{6})$
$Np = 4, Ns = (4)$	$D4 = 2D_{max} + 2(R_e + A)\cos(\frac{pi}{Ns})\sin(\frac{3\pi}{8})$
$Np = 4, Ns = (6)$	$D4 = D_{max} + 6(R_e + A)\sin(\frac{pi}{Ns})$
$Np = 4, Ns = (8)$	$D4 = D_{max} + 8(R_e + A)\sin(\frac{pi}{Ns})$
$Np = 4, Ns = (10, 12)$	$D4 = 2D_{max} + 6(R_e + A)\cos(\frac{pi}{Ns})\sin(\frac{\pi}{8})$
$Np = 5, Ns = (4)$	$D5 = 2D_{max} + 2(R_e + A)\cos(\frac{pi}{Ns})\sin(\frac{2\pi}{5})$
$Np = 5, Ns = (6)$	$D5 = D_{max} + 6(R_e + A)\sin(\frac{pi}{Ns})$
$Np = 5, Ns = (8)$	$D5 = D_{max} + 8(R_e + A)\sin(\frac{pi}{Ns})$
$Np = 5, Ns = (10)$	$D5 = D_{max} + 10(R_e + A)\sin(\frac{pi}{Ns})$
$Np = 5, Ns = (12)$	$D5 = 2D_{max} + 8(R_e + A)\cos(\frac{pi}{Ns})\sin(\frac{\pi}{10})$

Proof: The distance expressions are valid when satellites are $\frac{\pi}{Ns}$ radians from the x = 0 plane (measured in the z = 0 plane). Source and destination are in the x = 0 plane and midway between two orbital planes. Both source and destination are at least $D_{max}/2$ away from all satellites. The satellite-to-satellite paths must be one of the following segment types:

1. same orbit plane: $distance = 2(R_e + A)\sin(\frac{m\pi}{Ns})$: with $m = 1, 2, \ldots$ and $distance \leq \frac{D_{max}}{2}$

2. different orbit planes but in the same y-z plane:
   $distance = 2(R_e + A)\sin(Az)\sin(\frac{m\pi}{2Np})$;
   with $m = 1, 2, \ldots$, $distance \leq \frac{D_{max}}{2}$, and $0 \leq Az \leq \pi$

3. different orbit planes and in different y-z planes (a diagonal path): $distance \leq \frac{D_{max}}{2}$

The closed formulas of Theorem 3.3 are the sum of D_{max} and some combination of the three satellite-to-satellite segment types.

The closed formulas can be shown to be a local maximum by noting that a change in source-destination location will provide a shorter path (there are only 2 Np points which are $D_{max}/2$ away from all satellites). Also a change in satellite position will require the source-to-satellite and destination-to- satellite distance to be less than $D_{max}/2$ and the diagonally opposite satellite distance will become shorter (less than D_{max}). In the case of $Np = 2$ the derivative of the satellite path when $\mathrm{Az} = \frac{\pi}{2} - \frac{\pi}{Ns}$ is:

$$\begin{aligned}
\text{satellite path} &= \sqrt{2}(R_e + A)\sin(Az) + Diag \text{ where} \\
Diag^2 &\equiv 2(R_e + A)^2(1 - \cos(Az)\cos(Az + \frac{2\pi}{Ns})) \\
\text{path derivative} &= \sqrt{2}(R_e + A)\cos(Az) + \frac{D1}{D2} \text{ where} \\
D1 &\equiv \frac{\sqrt{2}(R_e + A)}{2}(\cos(Az)\sin(Az + \frac{2\pi}{Ns}) + \sin(Az)\cos(Az + \frac{2\pi}{Ns})) \\
D2 &\equiv \sqrt{1 - \cos(Az)\cos(Az + \frac{2\pi}{Ns})}
\end{aligned}$$

Substituting $\mathrm{Az} = \frac{\pi}{2} - \frac{\pi}{Ns}$ makes the satellite path derivative $\sqrt{2}(R_e + A)\sin(\frac{\pi}{Ns})$. The derivative of the source-to-satellite path is:

$$\begin{aligned}
\text{source-to-satellite path} &= D \text{ where} \\
D^2 &\equiv R_e^2 + (R_e + A)^2 - \sqrt{2}(R_e + A)Re\sin(Az)
\end{aligned}$$

Taking the derivative of D with respect to Az

$$D' = -\frac{\sqrt{2}}{2}(R_e + A)R_e \cos(Az)/D \text{ or}$$
$$D' = \sqrt{2}(R_e + A)R_e \sin(\frac{\pi}{Ns})/D_{max}$$

with substitution $\mathrm{Az} = \frac{\pi}{2} - \frac{\pi}{Ns}$. Hence the rate of change for source and destination is $2D'$. The source-to-destination geodesic will increase as $\mathrm{Az} = \frac{\pi}{2} - \frac{\pi}{Ns}$ is approached. The derivative will not be zero at the local maximum because the local maximum exists because of a boundary condition: the source-to-satellite distance equation is only valid to $\mathrm{Az} = \frac{\pi}{2} - \frac{\pi}{Ns}$. The diagram below shows the four diagonal satellites and location of the source for $\mathrm{Az} > \frac{\pi}{2} - \frac{\pi}{Ns}$. Satellites 1,3 are moving apart at a slower rate than satellites 2,4 are moving together, hence the source must move at the rate of satellites 2,4 in order to stay $D_{max}/2$ away from satellites 2,4. The combination of a shorter diagonal distance, shorter source to satellite distance, and shorter destination to satellite offsets the increase in distance of satellites 1,3, thus producing a monotonically decreasing geodesic distance for $\mathrm{Az} > \frac{\pi}{2} - \frac{\pi}{Ns}$.

Since the closed formulas are for local maximums, to show they are for global maximums requires Algorithm 2. The three-dimensional search space is:

1. Azimuth of source (destination)

2. Elevation of source (destination)

3. Satellite displacement angle, Az(t)

Use Algorithm 2 outside the vicinity of the local maximum with a grid size such that:

$$\text{upper bound error} < \text{closed formula distance} - DO$$

where DO is some minimum in the vicinity of the local maximum.

Assume adjacent pairs of a grid have the same satellite path. If a longer geodesic exists, then the sum of the maximum distance of the four corners and the upper bound error must exceed the closed formula distance. If adjacent pairs have different satellite paths, then the circles of coverage need to be solved to ensure some longer path is not missed. Since no longer path can be found for the stated topologies, the closed forms of theorem III are the maximum geodesic distances for s = $R_e\pi$.

Corollary: D2 in theorem 3.3 is valid for $Ns > 12$ because it is true for $Ns < 12$ and as Ns increases, the degree of satellite coverage overlap increases. This is shown by comparing the distance between adjacent circles of coverage with the diameter of the circles. If a longer geodesic existed than D2, then it would exit at smaller values of Ns.

3.5 Node Connectivity

The $Np.Ns$ topologies have at least four node disjoint paths at the minimum altitude for complete coverage. Table 3.1 contains the minimum and time average node connectivities. These connectivities were obtained using a topology generator program and applying Even's algorithm [17] whenever the satellite adjacency matrix changed.

The proof that four node disjoint paths exist can be made by assuming a mesh link assignment as follows:

1. Let the satellites be rotated about the x-axis, then the satellite positions are defined by:

$$
\begin{aligned}
X_{ij} &= (Re + A)\cos\left(\frac{2\pi j}{N_s} + Az\right) \\
Y_{ij} &= (Re + A)\sin\left(\frac{2\pi j}{N_s} + Az\right)\cos\left(\frac{\pi i}{N_p}\right) \\
Z_{ij} &= (Re + A)\sin\left(\frac{2\pi j}{N_s} + Az\right)\sin\left(\frac{\pi i}{N_p}\right)
\end{aligned}
$$

Table 3.1. Node Connectivity for $Np.Ns$ Topologies

Two Planes at 90 degrees					Three Planes at 60 degrees				
$N_p * N_s$ satellites		node k-conn			$N_p * N_s$ satellites		node k-conn		
		alt.	min.	avg.			alt.	min.	avg.
2X3 = 6		11662	5	5.0	3X3 = 9		8351	7	7.8
2X4 = 8		6378	4	4.0	3X4 = 12		4037	6	6.0
2X5 = 10		4771	4	4.0	3X5 = 15		2725	6	6.0
2X6 = 12		4037	4	4.0	3X6 = 18		2126	5	6.0
2X7 = 14		3633	4	4.0	3X7 = 21		1796	6	6.0
2X8 = 16		3385	8	8.0	3X8 = 24		1593	5	6.0
2X9 = 18		3221	8	8.0	3X9 = 27		1459	5	6.0
Four Planes at 45 degrees					Five Planes at 36 degrees				
$N_p * N_s$ satellites		node k-conn			$N_p * N_s$ satellites		node k-conn		
		alt.	min.	avg.			alt.	min.	avg.
4X3 = 12		7429	7	9.2	5X3 = 15		7034	7	10.1
4X4 = 16		3385	7	8.0	5X4 = 20		3106	8	9.9
4X5 = 20		2155	6	7.9	5X5 = 25		1911	6	9.3
4X6 = 24		1593	5	7.5	5X6 = 30		1366	7	8.5
4X7 = 28		1284	6	7.0	5X7 = 35		1065	7	7.0
4X8 = 32		1094	5	7.2	5X8 = 40		881	7	7.0
4X9 = 36		969	6	7.0	5X9 = 45		759	6	7.0

Notes:

1. Averages are time averages
2. Altitude in kilometers
3. The N_p, N_s systems provide complete coverage of the earth at all times

where $i = 0, 1, \ldots, N_p - 1$ and $j = 0, 1, \ldots, N_s - 1$

2. The distance, D between visible satellites i,j and i', j' is:

$$\begin{aligned} D &= (Re + A)\sqrt{2(1-c)}, \text{ where} \\ c &= \cos\left(\frac{2\pi j}{N_s} + Az\right)\cos\left(\frac{2\pi j'}{N_s} + Az\right) \\ &\quad + \sin\left(\frac{2\pi j}{N_s} + Az\right)\sin\left(\frac{2\pi j'}{N_s} + Az\right)\cos\left(\frac{\pi(i-i')}{N_p}\right) \end{aligned} \tag{3.8}$$

3. Satellite i,j is connected to four satellites as follows:

decreasing i:	if i > 0	(i-1),j
	if i = 0 and N_s odd	(N_p-1),(j-1) mod N_s
	if i = 0 and N_s even	(N_p-1),(N_s-1-j)
increasing i:	if i < N_p-1	(i+1),j
	if i = N_p-1 and N_s odd	0,(j+1) mod N_s
	if i = N_p-1 and N_s even	0,(N_s-1-j)
decreasing j:	i,(j-1) mod N_s	
increasing j:	i,(j+1) mod N_s	

The link assignment is valid because each satellite has four links and the maximum distance links are $\leq D_{max}$. The latter is true because the maximum distance links are between diagonally opposite satellites, which is constrained by the minimum altitude for complete coverage. The four links are true because of the four cases for $Az < \frac{2\pi}{N_s}$. Az can be constrained to this interval because it is the period of the topology.

To prove the given link assignment is four connected, an inductive proof can be used. There are two basis topologies, odd and even N_s. Appendix A has the node

disjoint paths for $Np = 2, Ns = 3$ and $Np = 2, Ns = 4$ which were found by Max Flow. Tables A.2 and A.4 show the four node disjoint paths for all pairs of satellites. The inductive step is similar to that for the three, time invariant node disjoint paths and will be given for that proof.

The time invariant link assignment is the set of links which do not require retargeting. The link assignment is as follows:

decreasing i:	if i > 0	(i-1).j
increasing i:	if i < N_p-1	(i+1).j
decreasing j:	i.(j-1) mod N_s	
increasing j:	i.(j+1) mod N_s	

Each satellite on planes i = 0 and i = N_p - 1 have three antennas and all other satellites have four antennas. The basis topology is $Np = 2, Ns = 3$ which is three connected. Table A.6 shows the three node disjoint paths for all pairs of satellites.

For the inductive step consider adding an internal plane $0 < i_p < N_p - 1$. Each satellite on i_p has four neighbors. By the link assignment, a mesh exists for any i_p. Since the connectivity of the mesh is four, then the connectivity of each satellite is at least three. Next consider adding an additional satellite to all planes $0 < j < N_s - 1$. By the link assignment each satellite has three neighbors for $i = 0, N_p - 1$ and assume it is three connected. Adding satellite j will not reduce the connectivity because it does not change the structure (i.e., each satellite has a left and right neighbor on the same plane and one neighbor on an adjacent plane).

3.6 Optimizing the Delay and Connectivity

A topology of N satellites at minimum altitude can be analyzed to provide optimum values for Np and Ns ($N = Np * Ns$) such that a specified delay metric is minimized. The delay metric could be maximum propagation delay, average maximum propagation delay (over time), etc. Suppose the metric is the maximum propagation delay for maximally separated terrestrial points at minimum altitude. The equations from section 3.4 can be used for Ns even; for Ns odd, simulation is required. Table 3.2 contains the propagation delays for $Np = [2,3,4,5]$ and $Ns = [3,4,5,6,7,8,9]$. The minimum altitude for systems with phase offset of $2\pi/(Np * Ns)$ is also given for comparison purposes. These altitudes were determined iteratively. Notice that for $Ns > Np$ there is little or no altitude reduction by using a non-zero phase offset. Also note that the lowest altitude for $N = Np * Ns$ satellites occurs when $Ns \geq Np$ and Ns is odd. This is because the elevation angle spans 180 degrees, while the azimuth angle spans 360 degrees. Hence Ns should be about twice as large as Np and odd values of Ns ensure the propagation path does not overshoot either source or destination. For Ns even, the propagation path overshoots either source or destination for furthest terrestrial points, resulting in a longer delay.

Optimizing connectivity for an Np,Ns topology can be performed in a manner similar to that used for propagation delay. However, unlike propagation delay where minimum altitude is desired, connectivity improves with increasing altitude. Hence some maximum tolerable delay must be specified and thereby some maximum altitude for the satellite topology.

Table 3.2. Minimum Altitude and Maximum Propagation Delay for Furthest Terrestrial Points

Two Planes at 90 degrees				Three Planes at 60 degrees			
$N_p * N_s$ satellites	phase=0		phase=P_n	$N_p * N_s$ satellites	phase=0		phase=P_n
	alt.	delay	alt.		alt.	delay	alt.
2X3 = 6	11662	205	11662	3X3 = 9	8351	160	8131
2X4 = 8	6378	190	5897	3X4 = 12	4037	152	3940
2X5 = 10	4771	139	4771	3X5 = 15	2725	106	2710
2X6 = 12	4037	152	3857	3X6 = 18	2126	123	2090
2X7 = 14	3633	131	3633	3X7 = 21	1796	97	1785
2X8 = 16	3385	141	3291	3X8 = 24	1593	113	1522
2X9 = 18	3221	127	3221	3X9 = 27	1459	96	1455
Four Planes at 45 degrees				Five Planes at 36 degrees			
$N_p * N_s$ satellites	phase=0		phase=P_n	$N_p * N_s$ satellites	phase=0		phase=P_n
	alt.	delay	alt.		alt.	delay	alt.
4X3 = 12	7429	148	7320	5X3 = 15	7034	141	5695
4X4 = 16	3385	141	3350	5X4 = 20	3106	136	3085
4X5 = 20	2155	96	2140	5X5 = 25	1911	91	1895
4X6 = 24	1593	112	1580	5X6 = 30	1366	107	1360
4X7 = 28	1284	86	1273	5X7 = 35	1065	83	1055
4X8 = 32	1094	102	1090	5X8 = 40	881	97	875
4X9 = 36	969	85	965	5X9 = 45	759	79	755

Notes:

1. $P_n = 2\pi/(N_p * N_s)$
2. Altitude in kilometers
3. Delay in milliseconds between two terrestrial locations separated by πR_e
4. The N_p, N_s systems provide complete coverage of the earth at all times

CHAPTER 4
ALGORITHMS FOR COMPUTING PROPAGATION DELAY

Determining performance measures for general topologies can be done with approximate analytic solutions or more precisely with simulation. For those who need precise solutions, two algorithms and their correctness proofs are given in the following sections. These algorithms measure propagation delay.

4.1 Minimum Propagation Delay Algorithm

Definition of Algorithm 1:

Determine the minimum propagation delay between arbitrary source and destination locations on the earth separated by terrestrial arc distances, $S = \{s_1, s_2, \ldots, s_N\}$ using a single altitude topology.

General Description:

The algorithm uses dynamic programming and maintains the shortest propagation distances between arbitrary points on the earth for each pair of satellites for one orbit period. The algorithm works because:

1. The source, destination, and terminating satellites are coplanar for the shortest geodesic

2. The ratio of terrestrial arc distance to propagation path distance is monotonically decreasing as the angle of satellite separation increases from $\frac{s}{R_e} - 2\theta$ to $\frac{s}{R_e} + 2\theta$

Every possible pair of satellite-to-satellite paths will be tested to determine whether they can offer a shorter path for some terrestrial arc distance in S. The algorithm

uses a closed formula for the propagation path distance as a function of terrestrial arc distance. Thus eliminating grid search for the minimum. In addition, a data structure is maintained which eliminates the need to test all terrestrial distances in S.

Pseudo-Code for Algorithm 1:

1. Initialize
   (a) Choose a set of terrestrial arc distances $S = \{s_1, s_2, \ldots, s_N\}$ and let the index into the set be I_s, $I_s = 1, \ldots, N$
   (b) Initialize arrays MinDistance[] and TerrestrialAngle[] for $I_s = 1, \ldots, N$ for corresponding terrestrial distances $s_1, \ldots, s_N$

$$\begin{aligned} \text{MinDistance}[I_s] &= MAXINT \ \{\text{for terrestrial distance } s_i \geq 2R_e\theta\} \\ &= 2\sqrt{A'^2 + 2R_e(R_e + A')(1 - \cos\left(\frac{s_i}{2R_e}\right))} \\ &\quad \{\text{for } s_i < 2R_e\theta\} \\ \text{TerrestrialAngle}[I_s] &= s_i/R_e \ \{\text{angle of terrestrial points }\} \end{aligned}$$

   (c) $A_z = 0$ {simulation angle is 0 to 2π}
   (d) N_{sat} = {number of satellites}
   (e) $m = N_{sat} * (N_{sat} - 1)/2$ { number of geodesics }

2. At simulation angle A_z, compute the distance between all pairs of satellites at different locations and store the distances less than D'_{max} in a matrix. Use Floyd's all-pairs shortest-path algorithm to compute the geodesic distance between all satellite pairs. Label the m geodesic paths P_1 to P_m and corresponding angles of separation A_1 to A_m.

3. **for** $i = 1$ **to** m **do begin**

4. {For each A_i, compute the maximum terrestrial distance, S_{max}}
   $S_{max} = (A_i + 2\theta)R_e$

   if $S_{max} \leq \pi R_e$
   then path_distance = Pi_Distance { path P_i geodesic distance} + D'_{max}
   else begin { $S_{max} > \pi R_e$, so source distance < $D'_{max}/2$ }
   path_distance = Pi_Distance + $\sqrt{A'^2 + 2R_e(R_e + A')(1 - \cos((\pi - A_i)/2))}$
   $S_{max} = \pi R_e$
   end
   k = { index of S_{max} }
   {update all shorter geodesics for path P_i}
   while (MinDistance[k] > path_distance) and ($k > 0$)

and (TerrestrialAngle[k] > A_i)
do begin
MinDistance[k] = path_distance; $k = k - 1$
q = TerrestrialAngle[k] - A_i { q is a temporary variable }
path_distance = Pi_Distance + $\sqrt{A'^2 + 2R_e(R_e + A')(1 - \cos(q/2))}$
end
end i

5. Increment A_z and go to step 2 if $A_z < 2\pi$.

Proof of Algorithm: The algorithm has three loops:

1. Outer loop for all satellite azimuth positions ($0 \leq Az \leq 2\pi$)

2. Inner loop for geodesics P1 to Pm

3. While loop to update MinDistance[k] for each geodesic Pi

Outer Loop - All possible satellite positions are considered because the topology has period 2π. This assumes an Az(t) step approaching zero. In practice, a finite Az(t) step will be used which will make the MinDistance values in error. An upper bound on the error can be derived and is given as:

$$error\_distance = 2(R_e + A)DelAz(M + DelAz\frac{R_e}{A})$$

where DelAz and M are the azimuth step and number of satellites on a path respectively.

Lemma 3.1 can be used to determine which satellites are visible. Floyd's all-pairs shortest path algorithm will provide the geodesic distance between all pairs of satellites.

Inner Loop - All possible geodesics are considered. For Ai < 2θ, array MinDistance[] has the shortest delay because these angles require only 1 satellite.

While Loop - Prove the algorithm segment for step 4.

if $S_{max} \leq \pi R_e$
then path_distance = Pi_Distance { path P_i geodesic distance} + D'_{max}
else begin { $S_{max} > \pi R_e$, so source distance < $D'_{max}/2$ }
path_distance = Pi_Distance + $\sqrt{A'^2 + 2R_e(R_e + A')(1 - \cos((\pi - A_i)/2))}$
$S_{max} = \pi R_e$
end

Variables S_{max} and *path_distance* have the correct values after executing the if statement because:

1. if $S_{max} \leq R_e\pi$ holds before the if statement then the path distance is the satellite path + $2D'_{max}/2$

2. if $S_{max} > R_e\pi$ holds before the if statement then S_{max} is limited to $R_e\pi$ (the maximum source-to-destination arc distance) and the source-to-satellite distance is less than $D'_{max}/2$ and is given by

$$\sqrt{A'^2 + 2R_e(R_e + A')(1 - \cos((\pi - A_i)/2))}$$

Pre-conditions of the while loop:

1. k has index of S_{max}
2. MinDistance[] has the minimum path lengths for Is = 1 to N for the geodesics tested so far

k = { index of S_{max} }
{update all shorter geodesics for path P_i}
while (MinDistance[k] > path_distance) and ($k > 0$)
and (TerrestrialAngle[k] > A_i)
do begin
MinDistance[k] = path_distance; $k = k - 1$
q = TerrestrialAngle[k] - A_i { q is a temporary variable }
path_distance = Pi_Distance + $\sqrt{A'^2 + 2R_e(R_e + A')(1 - \cos(q/2))}$
end

Post-conditions of the while loop:

1. k has index for which a shorter path existed

2. MinDistance[] has the minimum path lengths for Is = 1 to N for the geodesics tested so far

Loop Invariant: MinDistance[i] = MINIMUM {MinDistance[i].path_distance } for k < i ≤ {index of S_{max} }, where MinDistance[k] > path_distance

1. the upper bound of i follows from the pre-condition

2. the lower bound of i follows since the ratio of terrestrial distance to propagation path distance is monotonically decreasing for decreasing terrestrial distance; hence if MinDistance[i] > path_distance, then all valid smaller values of i can be ignored because some other previous path had a shorter satellite to satellite path (i.e., a larger ratio of terrestrial distance to propagation path distance)

The running time of Algorithm 1 is O[(S_n^3 + m)steps]; where

S_n = number of satellites,

steps = number of azimuth steps,

m = (number of MinDistance updates) $\times S_n(S_n - 1)/2$.

The number of MinDistance updates is topology dependent and varies from 1 to S_n^2.

4.2 Time Average and Maximum Propagation Delay Algorithm

Definition of Algorithm 2:

Compute the time average and maximum propagation delay for a set of arbitrary terrestrial locations separated by the terrestrial arc distance $R_e\pi$ using a single altitude topology.

General Description:

The algorithm computes the geodesics between a set of source-destination pairs by first computing the geodesics between all satellite pairs and then comparing all possible paths between source and destination for the shortest geodesic distance. This algorithm becomes more efficient for low altitudes where only a small number of the total satellites are over head at one time.

Pseudo-Code for Algorithm 2:

1. Initialize

   (a) $A_z = 0$ {simulation angle is 0 to 2π }

   (b)
   ```
   for i = 1 to P do begin
           {P = number of source-destination pairs}
           TotalDistance[i] = 0 {running sum of path length}
           MaxDistance[i] = 0 {maximum propagation path length}
   end
   ```

   (c) S[i] = {terrestrial grid points for source, i = 1 to P}

   (d) D[i] = {terrestrial grid points for destination}

   (e) N_{sat} = {number of satellites}

2. At simulation angle A_z, compute the distance between all-pairs of satellites at different locations and store the distances less than D'_{max} in a matrix. Use Floyd's algorithm to compute the geodesic for all satellite pairs. The result is in array Dis[], with element Dis[k,m] the geodesic from satellite k to satellite m.

3.
   ```
   for i = 1 to P do begin
           {Determine which satellites are visible to the source and destination}
           Sindex = 0 {number of satellites visible to the source}
           Dindex = 0 {number of satellites visible to the destination}
           MINd = MAXINT; {minimum geodesic distance}
           for k = 1 to N_sat
           do begin
                   DS[k] = {distance between S[i] and satellite k}
                   if DS[k] < D_max/2
                   then begin
                           Sindex = Sindex + 1;    Snode[Sindex] = k
                   end
                   DD[k] = {distance between D[i] and satellite k}
                   if DD[k] < D_max/2
                   then begin
   ```

```
                    Dindex = Dindex + 1:    Dnode[Dindex] = k
            end
        end     {Determine the shortest geodesic between all possible paths}
        for k = 1 to Sindex do
            for m = 1 to Dindex do
                MINd = min ( MINd,
                        DS[Snode[k]]+DD[Dnode[m]]+Dis[Snode[k],
                        Dnode[m]] )
        TotalDistance[i] = TotalDistance[i] + MINd {update for average distance}
        MaxDistance[i] = MAX(MaxDistance[i],MINd) {update for max distance}
end i
```

4. **if** $A_z < 2\pi$, **then** increment A_z and go to step 2.
5. Average_Distance[i] = TotalDistance[i]/[P* number_of_azimuth_steps]

Proof of Algorithm: The algorithm has four loops:

1. Outer loop for $0 \leq \mathrm{Az} \leq 2\pi$
2. Inner loop using index i for set of grid points
3. Visibility loop
4. Shortest path loop

Outer Loop - All possible satellite positions are considered because the topology has period 2π. This assumes an Az(t) step approaching zero. In practice, a finite Az(t) step will be used which will make the propagation distances in error. An upper bound on the error can be derived and is given as:

$$error\_distance = (R_e + A)DelAz(2M + (DelG \times DelG + DelAz)\frac{R_e}{A})$$

where DelAz, DelG and M are the azimuth step, grid angle, and number of satellites on a path respectively.

Lemma 3.1 can be used to determine which satellites are visible. Floyd's all-pairs shortest path algorithm will provide the geodesic distance between all-pairs of satellites.

Inner Loop - All grid points are considered.

Visibility Loop - Prove the program segment for step 3.

Pre-conditions: arrays Snode and Dnode are empty

```
for i = 1 to P do begin
        {Determine which satellites are visible to the source and destination}
        Sindex = 0 {number of satellites visible to the source}
        Dindex = 0 {number of satellites visible to the destination}
        MINd = MAXINT; {minimum geodesic distance}
        for k = 1 to N_sat
        do begin
                DS[k] = {distance between S[i] and satellite k}
                if DS[k] < D_max/2
                then begin
                        Sindex = Sindex + 1;    Snode[Sindex] = k
                end
                DD[k] = {distance between D[i] and satellite k}
                if DD[k] < D_max/2
                then begin
                        Dindex = Dindex + 1;    Dnode[Dindex] = k
                end
        end
```

Post-conditions: Snode[] has the set of satellites visible to source S[i] and Dnode[] has the set of satellites visible to destination D[i]

Pre-condition: MINd = MAXINT

```
        {Determine the shortest geodesic between all possible paths}
        for k = 1 to Sindex do
                for m = 1 to Dindex do
                        MINd = min ( MINd,
                                DS[Snode[k]]+DD[Dnode[m]]+Dis[Snode[k],
                                Dnode[m]] )
        TotalDistance[i] = TotalDistance[i] + MINd {update for average distance}
        MaxDistance[i] = MAX(MaxDistance[i],MINd) {update for max distance}
```

Post-condition: MINd = MINIMUM { S[i] to D[i] path distance }

Loop Invariant: MINd = MINIMUM { S[i] to D[i] path distance, MINd }

All possible satellite paths are tested between S[i] and D[i]. The minimum distance is in MINd at loop termination.

The running time of Algorithm 2 as compared to Dijkstra's single source algorithm is:

Algorithm A.2	Dijkstra
$O(L(2N + M^2))$	$O(L(N+2)^2)$

L = (number of azimuth steps) * (number of source-destination pairs)

M = average number of visible satellites from the earth

N = number of satellites in the network

Note: (1) Assumes (number of source-destination pairs) $*(2N + M^2) \gg N^3$

(2) $M \rightarrow N/2$ as $A \rightarrow \infty$

CHAPTER 5
ANALYSIS OF A 2-LEVEL HIERARCHICAL MODEL

An analytic approximation for the total average delay of a point-to-point satellite network with arbitrary topologies can be obtained by using a parameterized 2-level hierarchy. The advantages of such a network are a reduction in the complexity of delay and routing analysis. The parameters include link capacity, traffic flow between all pairs of satellites, number of satellites, number of antennas per satellite, and two altitudes. The link assignment provides a 2-level hierarchy by using a backbone (level-1) and local regions (level-2). The routing strategy uses traffic flow balancing by splitting traffic between node disjoint paths.

Satellites are used in a backbone which connects regions of diameter D_{max} on each arbitrary altitude shell and also between the two shells. The local regions have diameter D_{max} and form a "wrapped-around mesh" which offers a region disjoint structure for routing over four disjoint paths. The diameter of the regions ensures all satellites within the region are visible to one another.

5.1 Definition and Description of the Model

1. All satellites are contained in a circular region with diameter D_{max}. The regions cover all points on a shell.

2. The centers of the regions are:

$$x_{ij} = (Re + A)\sin\left(\frac{2\pi j}{N_s} + az\right)\cos\left(\frac{\pi i}{N_p}\right)$$

$$y_{ij} = (Re + A)\sin\left(\frac{2\pi j}{N_s} + az\right)\sin\left(\frac{\pi i}{N_p}\right)$$

$$z_{ij} = (Re + A)\cos\left(\frac{2\pi j}{N_s} + az\right)$$

where $i = 0, 1, \ldots, N_p - 1$ and $j = 0, 1, \ldots, N_s - 1$

3. The minimum altitude of the N_p, N_s region model is derived in the next section, and given as:

$$\begin{aligned}
A_{min} &= \left(\frac{1}{c} - 1\right) R_e, \text{ where} \qquad (5.1)\\
c &= \sin(El)\sin(El')\cos\left(\frac{\pi}{2N_p}\right) + \cos(El)\cos(El')\\
ang &= \lfloor \frac{N_s}{4} \rfloor \frac{2\pi}{N_s} + az, \text{ where } 0 \leq az < \frac{2\pi}{N_s}\\
ang' &= ang \text{ if } ang < \frac{\pi}{2}\\
&= ang - \frac{2\pi}{N_s}, \text{ otherwise}\\
El &= ang' \text{ if } \frac{\pi}{2} - ang' < ang' + \frac{2\pi}{N_s} - \frac{\pi}{2}\\
&= ang' + \frac{2\pi}{N_s}, \text{ otherwise}\\
El' &= -\arctan\left(\frac{d}{e}\right) \qquad \left(El < El' < El + \frac{2\pi}{N_s}\right)\\
d &= \cos(El) - \cos\left(El + \frac{2\pi}{N_s}\right)\\
e &= \cos\left(\frac{\pi}{2N_p}\right)\left(sin(El) - \sin\left(El + \frac{2\pi}{N_s}\right)\right)
\end{aligned}$$

$N_p \geq 2$ and $N_s \geq 3$

4. The number of regions is $N_p * N_s$ for $az > 0$.

5. The link assignment in a region is made independently of propagation distance between satellites.

6. The propagation distance between satellites in the circular region is a random variable. The average satellite-to-satellite distance is $D_{avg}D_{max}/2$, where D_{avg}

is a constant which depends only on the altitude of the satellite shell. The value of D_{avg} ranges from 1.0 at 10000 km to 0.93 at 1000 km.

7. Satellites on a shell can have an arbitrary topology.

For $az = 0$ and N_s even there are N_p regions for j = 0 and j = $N_s/2$. Hence the set of regions do not form a region disjoint structure. However, at the higher altitude, A'_{min}, the regions for j = 1 and j = N_s - 1 will cover all points of the N_p regions at j = 0. Similarly, the j = $N_s/2$ regions will also be covered. Hence the values of j = $1, 2, \ldots, \frac{N_s}{2} - 1, \frac{N_s}{2} + 1, \ldots, Ns - 1$ form a region disjoint structure. The equation for A'_{min} is derived in the next section, and given as:

$$A'_{min} = R_e\sqrt{1 + \tan^2\left(\frac{2\pi}{N_s}\right)} - R_e \tag{5.2}$$

For $N_p = 3$, $N_s = 10$, $az = 0$ there are 24 regions which cover a shell at $A > A'_{min} =$ 1506 km. A lower altitude is obtained by relaxing the equal spacing of the regions, i.e. modify the angle $\frac{2\pi j}{N_s}$ to $\frac{2\pi j}{N_s} - \Delta$, where Δ is large for $j = 0$ and $j = \frac{N_s}{2}$ and zero for $j = \frac{N_s}{4}$ and $j = 3\frac{N_s}{4}$. In addition, by allowing the regions to be 10% greater than D_{max} the 24 regions cover a shell at $A >$1000 km.

5.2 Minimum Altitude Derivation of Np,Ns Regions

The minimum altitude of the N_p, N_s region model is:

$$\begin{aligned}
A_{min} &= \left(\frac{1}{c} - 1\right) R_e, \text{ where} \\
c &= \sin(El)\sin(El')\cos\left(\frac{\pi}{2N_p}\right) + \cos(El)\cos(El') \\
ang &= \lfloor\frac{N_s}{4}\rfloor\frac{2\pi}{N_s} + az, \text{ where } 0 \leq az < \frac{2\pi}{N_s} \\
ang' &= ang \text{ if } ang < \frac{\pi}{2} \\
&= ang - \frac{2\pi}{N_s}, \text{ otherwise}
\end{aligned}$$

$$
\begin{aligned}
El &= ang' \text{ if } \frac{\pi}{2} - ang' < ang' + \frac{2\pi}{N_s} - \frac{\pi}{2} \\
&= ang' + \frac{2\pi}{N_s}, \text{ otherwise} \\
El' &= -\arctan\left(\frac{d}{e}\right) \qquad \left(El < El' < El + \frac{2\pi}{N_s}\right) \\
d &= \cos(El) - \cos\left(El + \frac{2\pi}{N_s}\right) \\
e &= \cos\left(\frac{\pi}{2N_p}\right)\left(sin(El) - \sin\left(El + \frac{2\pi}{N_s}\right)\right)
\end{aligned}
$$

Proof:

1. The distance squared between a region center (El, Az) and some arbitrary point (El', Az') on a sphere of radius R is:

$$
\begin{aligned}
D^2 &= 2R^2 - 2R^2 \sin(El)\sin(El')\cos(Az - Az') \\
&\quad -2R^2\cos(El)\cos(El'), \text{ where} \\
El &= az, \frac{2\pi}{Ns} + az, \ldots, (Ns-1)\frac{2\pi}{Ns} + az \\
Az &= 0, \frac{\pi}{Np}, \ldots, (Np-1)\frac{\pi}{Np}
\end{aligned}
$$

2. A point (El', Az') maximally distant from all region centers will be equally distant between the four adjacent region centers, hence

$$
\begin{aligned}
Az - Az' &= \frac{\pi}{2Np} \\
El' &= -\arctan\left(\frac{d}{e}\right), \text{ where} \\
d &= \cos(El) - \cos\left(El + \frac{2\pi}{Ns}\right) \\
e &= \cos\left(\frac{\pi}{2Np}\right)(\sin(El) - sin\left(El + \frac{2\pi}{Ns}\right))
\end{aligned}
$$

3. The region centers which are furthest apart are those nearest the z = 0 plane, hence $El = \frac{2\pi}{Ns}\lfloor\frac{Ns}{4}\rfloor + az$, $\frac{2\pi}{Ns}\lfloor\frac{Ns}{4}\rfloor + az - \frac{2\pi}{Ns}$, or $\frac{2\pi}{Ns}\lceil\frac{Ns}{4}\rceil + az - \frac{2\pi}{Ns}$ whichever is closest to $\frac{\pi}{2}$

4. Make the maximum distance squared, D^2 equal to $A^2 + D_{max}^2/4$

$$\begin{aligned} A^2 + D_{max}^2/4 &= 2R^2 - 2R^2 \sin(El) \sin(El') \cos(Az - Az') \\ &\quad -2R^2 \cos(El) \cos(El'), \text{ where} \\ R &= R_e + A \end{aligned}$$

5. The value of A which satisfies the equation is A_{min} and the desired expression is obtained.

The minimum altitude A'_{min} for which all points in j = 0 and j = $N_s/2$ are covered by adjacent regions for N_s even is:

$$A'_{min} = R_e \sqrt{1 + \tan^2\left(\frac{2\pi}{N_s}\right)} - R_e$$

Proof:

1. The angle between common orbit satellites j = 1 and j = N_s - 1 is $\frac{4\pi}{N_s}$.

2. Equate the maximum tangential angle, $\arctan\left(\frac{D_{max}}{2R_e}\right)$ to half the satellite separation angle and the expression follows.

5.3 Link Assignment

A comparison of propagation delay results using unlimited antennas for the topology types of N_p, N_s and random (equal distribution of satellites per unit area) shows only about a 15% difference. The metrics used in the comparison include average path delay and maximum propagation delay. In addition, the propagation delay for N_p, N_s topologies increases only about 10% when the number of antennas is reduced to four and a mesh link assignment is made. A mesh provides a node connectivity of

four with three, time invariant (no retargeting) node disjoint paths between all-pairs of satellites.

Table 5.1 contains the propagation delay results for geodesic paths using simulation for minimum altitude N_p,N_s topologies with unlimited antennae. Table 5.2 is for the same topologies, but using four antennas per satellite and a mesh link assignment. Both propagation delay (in milliseconds) and hop values are time averages.

The N_p,N_s mesh link assignment is defined as:

1. Let the satellites be rotated about the x-axis, then the satellite positions are defined by:

$$\begin{aligned} X_{ij} &= (Re + A)\cos\left(\frac{2\pi j}{N_s} + Az\right) \\ Y_{ij} &= (Re + A)\sin\left(\frac{2\pi j}{N_s} + Az\right)\cos\left(\frac{\pi i}{N_p}\right) \\ Z_{ij} &= (Re + A)\sin\left(\frac{2\pi j}{N_s} + Az\right)\sin\left(\frac{\pi i}{N_p}\right) \end{aligned}$$

where $i = 0, 1, \ldots, N_p - 1$ and $j = 0, 1, \ldots, N_s - 1$

2. The distance, D between visible satellites i,j and i', j' is:

$$\begin{aligned} D &= (Re + A)\sqrt{2(1-c)}, \text{ where} \\ c &= \cos\left(\frac{2\pi j}{N_s} + Az\right)\cos\left(\frac{2\pi j'}{N_s} + Az\right) \\ &\quad + \sin\left(\frac{2\pi j}{N_s} + Az\right)\sin\left(\frac{2\pi j'}{N_s} + Az\right)\cos\left(\frac{\pi(i-i')}{N_p}\right) \end{aligned} \tag{5.3}$$

3. Satellite i,j is connected to four satellites as follows:

decreasing i:	if i $>$ 0	(i-1),j
	if i = 0 and N_s odd	(N_p-1),(j-1) mod N_s
	if i = 0 and N_s even	(N_p-1),(N_s-1-j)

Table 5.1. Propagation Delay of Np,Ns Topologies. Unlimited Antennas

$N_p * N_s$ satellites	altitude (km)	distance mean	distance std_dev	distance max	hops mean	hops std_dev	hops max
2X3 = 6	11662	90.1	23.5	109.7	1.0	000.0	1.0
2X4 = 8	6378	68.9	27.1	118.3	1.2	0.4	2.0
2X5 = 10	4771	59.3	22.2	94.2	1.3	0.5	2.0
2X6 = 12	4037	56.2	22.9	101.1	1.5	0.5	2.5
2X7 = 14	3633	53.3	21.5	91.3	1.6	0.7	3.0
2X8 = 16	3385	51.0	21.2	92.0	1.4	0.5	2.0
2X9 = 18	3221	49.8	20.5	85.0	1.4	0.5	2.6
3X3 = 9	8351	70.4	23.6	99.6	1.0	0.2	1.8
3X4 = 12	4037	54.7	23.2	97.8	1.3	0.5	2.0
3X5 = 15	2725	47.6	19.1	76.9	1.5	0.5	2.0
3X6 = 18	2126	45.4	19.4	84.9	1.6	0.6	3.0
3X7 = 21	1796	43.4	18.2	75.5	1.8	0.7	3.0
3X8 = 24	1593	42.7	18.6	81.4	1.9	0.8	4.0
3X9 = 27	1459	41.7	18.1	75.2	2.1	1.0	4.0
4X3 = 12	7429	64.7	24.2	97.5	1.1	0.3	1.9
4X4 = 16	3385	50.3	22.4	92.0	1.3	0.5	2.2
4X5 = 20	2155	44.3	18.5	73.3	1.5	0.5	3.0
4X6 = 24	1593	42.3	18.6	79.7	1.7	0.7	3.0
4X7 = 28	1284	40.5	17.5	70.9	1.9	0.8	3.1
4X8 = 32	1094	39.8	17.7	76.3	2.1	0.9	4.0
4X9 = 36	969	39.1	17.2	70.5	2.2	1.0	4.0
5X3 = 15	7034	62.1	24.6	97.2	1.1	0.3	1.9
5X4 = 20	3106	48.5	22.2	89.4	1.4	0.5	2.5
5X5 = 25	1911	42.6	18.2	72.0	1.5	0.5	2.9
5X6 = 30	1366	40.8	18.3	77.5	1.8	0.7	3.5
5X7 = 35	1065	39.2	17.2	69.2	2.0	0.8	4.0
5X8 = 40	881	38.6	17.3	74.2	2.2	0.9	4.0
5X9 = 45	759	37.9	16.9	68.7	2.3	1.0	4.1

Table 5.2. Propagation Delay Using $Np.Ns$ Mesh Link Assignment

$N_p * N_s$ satellites	altitude (km)	distance			hops		
		mean	std_dev	max	mean	std_dev	max
2X3 = 6	11662	100.3	33.7	173.6	1.2	000.4	2.0
2X4 = 8	6378	74.9	30.4	120.3	1.4	0.5	2.0
2X5 = 10	4771	67.6	27.1	126.4	1.7	0.7	3.0
2X6 = 12	4037	61.3	25.4	107.3	1.8	0.7	3.0
2X7 = 14	3633	61.5	26.1	114.9	2.2	1.0	4.0
2X8 = 16	3385	57.4	24.5	105.6	2.3	1.0	4.0
2X9 = 18	3221	59.0	25.7	109.4	2.7	1.3	5.0
3X3 = 9	8351	76.6	27.5	103.6	1.5	0.6	2.8
3X4 = 12	4037	61.0	26.2	101.7	1.8	0.7	3.0
3X5 = 15	2725	52.6	23.2	79.9	2.0	0.8	3.0
3X6 = 18	2126	50.1	21.1	87.1	2.3	0.9	4.0
3X7 = 21	1796	48.5	20.1	78.5	2.5	1.0	4.0
3X8 = 24	1593	46.9	19.9	83.3	2.6	1.1	5.0
3X9 = 27	1459	44.6	19.2	77.2	2.7	1.1	5.0
4X3 = 12	7429	71.6	27.3	105.6	1.9	0.7	3.5
4X4 = 16	3385	57.2	25.6	100.4	2.2	1.0	4.0
4X5 = 20	2155	54.2	13.4	81.3	2.3	1.0	4.5
4X6 = 24	1593	46.9	20.1	84.5	2.6	1.1	5.0
4X7 = 28	1284	45.4	19.4	75.7	2.8	1.2	5.5
4X8 = 32	1094	44.0	18.9	79.8	3.0	1.3	6.1
4X9 = 36	969	42.2	18.0	73.3	3.1	1.4	6.2
5X3 = 15	7034	69.2	27.6	106.2	2.4	1.0	3.9
5X4 = 20	3106	55.5	25.5	98.9	2.6	1.2	5.0
5X5 = 25	1911	50.7	22.3	77.1	2.8	1.2	5.5
5X6 = 30	1366	45.5	19.7	82.9	3.0	1.3	6.0
5X7 = 35	1065	44.3	19.3	73.3	3.2	1.4	6.5
5X8 = 40	881	42.8	18.6	78.1	3.3	1.4	7.1
5X9 = 45	759	41.7	17.9	71.8	3.3	1.4	7.4

increasing i:	if i < N_p-1	(i+1).j
	if i = N_p-1 and N_s odd	0.(j+1) mod N_s
	if i = N_p-1 and N_s even	0.(N_s-1-j)
decreasing j:	i.(j-1) mod N_s	
increasing j:	i.(j+1) mod N_s	

The four shortest node disjoint paths of the N_p, N_s topologies can be generated from geometry and by using equation 5.3, the propagation delay of the paths can be computed. Hence propagation delay of the N_p,N_s topologies can be used to approximate the delay between regions of the N_p,N_s region model at each altitude (low and high). The delay range between the low altitude shell, A_{low} and the high altitude shell, A_{high} is:

$$\begin{aligned} DS_{min} &= A_{high} - A_{low} \\ DS_{max} &= A_{low}\sqrt{1 + 2\frac{R_e}{A_{low}}} \\ &\quad + A_{high}\sqrt{1 + 2\frac{R_e}{A_{high}}} \end{aligned}$$

An average of DS_{min} and DS_{max} is a good approximation for the distance between the two shells when $A_{high} > 2A_{low}$ since DS_{min} approaches DS_{max} as A_{high} increases.

The mesh link assignment algorithm for arbitrary topologies is to first construct a backbone as follows:

1. Determine satellites in a region overlap

2. Pick one satellite in each overlap area and assign two antennas to each of two overlapping regions (if more than two regions overlap, choose regions which

have the fewest possible region-region links)

Note: the region-to-region link is inside a satellite

3. For all adjacent regions which do not have a link: assign one link between regions with priority of assignment based on fewest inter-region links

4. If each region is linked to four adjacent neighbors; the mesh is formed. If the mesh is not formed then find all satellites which are visible to each other and in different unconnected regions and assign the link.

Since each local region can potentially be fully connected because all satellites are visible to each other, remaining antennas can be assigned to optimize some metric consisting of connectivity and maximum hops. The links between shells are best assigned based on queueing delay, since there is very little difference in propagation delay when $A_{high} > 2A_{low}$.

A 2-level hierarchy using 24 regions was chosen to evaluate the propagation delay of the network using only four antennas per satellite. This hierarchy model has a minimum altitude of 1506 km ($N_p = 3, N_s = 10, az = 0$) which is an average altitude attributed to sensor satellites (typically 1000-2000 km) in the SDI architecture. To determine the performance of the 2-level hierarchical link assignment, satellites at an altitude of 1506 km were randomly located with probability proportional to the area of the shell. The all-pairs geodesic distances were computed assuming an unlimited number of antennas and using only four antennas. The unlimited antenna case provided the optimum delay. Table 5.3 gives the mean, standard deviation, and maximum path delays in milliseconds. The top half of the table is for satellites in all 24 regions, the bottom half has satellites in only 23 regions. Region i = 2, j = 8 was constrained to be void to simulate a disaster condition. The results show only about a 15% increase in propagation delay when the number of antennas is reduced

to four. In addition, a void region has little affect on the delay. This was expected because of the three node disjoint paths. The table shows only the cases for 25-100 satellites because there was no appreciable change in delay for 100-150 satellites.

Table 5.3. Propagation Delay Using 24 Regions. Satellite Altitude = 1506 km

number satellites	unlimited antennas mean	std_dev	max	4 antennas mean	std_dev	max
satellites in 24 regions						
25	43.4	23.8	95.0	48.2	24.0	100.9
50	38.8	8.4	80.1	43.4	8.5	84.8
75	38.1	5.5	79.0	43.1	5.6	83.0
100	38.0	4.0	78.6	42.8	4.2	82.9
satellites in 23 regions only						
25	43.2	23.9	94.5	48.3	24.3	102.1
50	38.9	8.4	81.2	43.4	8.7	85.0
75	38.2	5.6	79.1	43.2	5.5	84.4
100	37.9	4.1	78.7	43.0	4.4	83.6

5.4 Queueing Delay of the Model

The queueing delay of the network is obtained by using M/M/1 assumptions with some arbitrary traffic flow between all pairs of satellites and routing based on minimizing the maximum link flow. The N_p, N_s topologies have four node disjoint paths and by splitting the flow, flow balancing can be achieved. In fact, for the $N_p = 2$, $N_s = 3$ topology all links have the same flow when each satellite sends the same traffic flow to the other five satellites in the network.

For arbitrary topologies, the flow can be split between region disjoint paths. The total delay is the sum of the propagation delay and queueing delay.

5.5 Computing D_{avg}

D_{avg} is a constant which varies slightly with altitude and is the proportionality constant between $D_{max}/2$ and the expected distance of two random points in a region of diameter D_{max}. The computation of D_{avg} using numerical integration is as follows:

1. Use spherical coordinates to locate points at altitude A:

$$\begin{aligned} x &= (R_e + A)\sin(El)\cos(Az) \\ y &= (R_e + A)\sin(El)\sin(Az) \\ z &= (R_e + A)\cos(El) \end{aligned}$$

where $0 \le Az \le 2\pi$ and $0 \le El \le El_{max} = 2\arcsin\left(\frac{D_{max}}{4(R_e+A)}\right)$

2. The distance between two points (Az, El) and (Az', El') is

$$Dis = \sqrt{2}(R_e + A)\sqrt{1 - \sin(El)\sin(El')\cos(Az - Az') - \cos(El)\cos(El')}$$

3. Algorithm:

   (a) for i,j = [1,N_{az}]: $Az - Az' = \frac{2\pi(i-j)}{N_{az}}$

   (b) compute frequency of abs(i-j):
   num[0] = N_{az}
   num[k] = 2 (N_{az}-k), for k = [1,N_{az}-1]

   (c) sum all pairs distance:

```
sum = 0
for El,El'= [1,N_el]
        do for k = 0 to N_az-1
                do sum = sum + num[k]*Dis(k,El,El')
```

4. Running time is $O(N_{el}^2 N_{az})$, which is much better than storing $N_{el} * N_{az}$ points and then computing the all pairs distances: an $O(N_{el}^2 N_{az}^2)$ algorithm

Table 5.4 contains computed values of D_{avg} for N_p..N_s regions. Three methods of computation were used for comparison.

Table 5.4. D_{avg} Versus Altitude

Altitude (km)	D_{max} (km)	N_p	N_s	Random D_{avg}-1	Grid D_{avg}-2	N_p..N_s D_{avg}-3
10497	31246	2	3	1.014	1.016	0.971
4669	18040	2	4	0.970	0.974	0.923
3365	14729	3	4	0.961	0.960	0.912
2616	12683	3	5	0.950	0.950	0.895
2126	11250	3	6	0.945	0.944	0.894
1593	9564	4	6	0.936	0.937	0.888
1366	8783	5	6	0.932	0.934	0.885
1256	8392	4	7	0.930	0.932	0.880
1014	7472	4	8	0.925	0.928	0.878
830	6715	5	8	0.928	0.925	0.884
673	6014	5	10	0.916	0.923	0.876

Notes:

1. Random topologies have numbers of satellites proportional to surface area.
2. Grid uses satellites on a grid of approximately 0.25 degrees.
3. $N_p = 5$, $N_s = 18$ topology.

5.6 Importance of the Model

1. The propagation delay for strings, rings, etc. in a region can be compared with closed formulas.

topology	average furthest points distance	# antennas
string	$D_{avg}(Nodes-1)D_{max}/2$	2
ring	$D_{avg}\lfloor\frac{Nodes}{2}\rfloor D_{max}/2$	2
double ring	$D_{avg}\left(1+\lfloor\frac{Nodes-1}{4}\rfloor\right)D_{max}/2$	3

2. The propagation distance between regions for single altitude systems can be compared with 2-altitude systems by using closed formulas.

3. Large numbers of satellites do not make the model more complex and the model becomes more valid as the number of satellite changes increase.

4. The model is valid for link assignment based on queue length or connectivity.

5. The model approximates any 2-altitude deployed satellite configuration.

5.7 Visibility Probability

If satellites are distributed uniformly per unit area, the probability of two satellites being visible is only a function of altitude. This is useful for analytic appoximations for connectivity. Table 5.5 contains the visibility probability values for various altitudes. For comparison purposes three methods of computation were used.

5.8 Conclusions

The N_p, N_s model provides a regular topology for which analytic solutions can be obtained for the performance of a network. A link assignment strategy which takes

Table 5.5. Probability of Two Satellites Being Visible

Altitude	Visibility Probability Percentage			
(km)	Random	Phase = p_n	Phase = 0	Grid
11000	90.0	82.1	82.1	85.8
10000	87.8	80.9	81.0	84.0
9000	84.7	79.6	79.5	81.9
8000	83.5	77.9	77.6	79.3
7000	80.5	75.2	75.5	76.1
6000	76.9	70.2	70.3	72.2
5000	72.9	66.6	66.2	67.2
4000	65.9	60.7	61.1	60.9
3000	57.2	51.9	51.9	52.8
2000	46.3	43.6	43.8	43.0
1000	27.8	27.5	27.5	29.3

Notes:

1. Random topologies have numbers of satellites proportional to surface area.
2. Phase = 0 and phase = p_n are $N_p = 5$, $N_s = 18$ topologies which are time averaged over one period with zero phase and $2\pi/90$ phase offsets respectively.
3. Grid uses satellites on a grid of approximately 1 degree.

advantage of the regularity inherent in the model is introduced. In addition the N_p, N_s model can be extended to regions in which a 2-level hierarchical model can be used for random topologies. Although the total delay measures assume steady state conditions, the transient response can be approximated when propagation delays are long compared to the queueing delays for high data rate communications. Future research will use the transient response to analyze link failures, thus help prevent link saturation in a congestion avoidance algorithm.

CHAPTER 6
K-PATHS OF MINIMUM TOTAL COST

Redundant messages can be used in a network to reduce delays caused by retransmission. By using disjoint paths, the reliability of message transfer can be improved. In particular, routing algorithms proposed for SDI have used multiple paths for increased survivability [8]. Suurballe [35] has given an algorithm for a minimum total cost set of node disjoint paths. However, if K-paths are desired and K disjoint paths do not exist, then some nodes must be repeated.

In this chapter two algorithms are presented for finding K-paths of minimum total cost. Section 6.1 gives the algorithm for fewest repeated nodes or links, and section 6.2 gives an algorithm for finding disjoints paths of minimum total cost which uses a different metric than the Suurballe algorithm, but is faster.

6.1 Fewest Repeated Nodes or Links

Let G be an undirected graph containing m links (edges) of non-negative cost and n nodes (vertices). An algorithm is presented which finds K-paths on G between a pair of nodes which are of minimum total cost and have fewest repeated nodes or links. The running time for dense graphs is $O(Kn^2)$ and for sparse graphs $O(Km \log n)$. The algorithm uses the solution to a modified minimum cost flow problem to find the desired K-paths.

6.1.1 Definitions

The path cost definition provides for metrics of propagation delay ($Link\_distance$) and retargeting frequency ($Link\_cost = 1/Remaining\_link\_time$). The latter metric

is for rapidly changing network topologies such as the proposed SDI communications network.

The path cost is:

$$
\begin{aligned}
&\text{path cost} && = \mid P \mid + W \sum_i Frequency_i \\
&\text{where } \mid P \mid && = \sum_i Link\_distance_i \text{ on path P} \\
& && \text{or} \\
& && = \text{MAX}[Link\_cost_i] \text{ on path P} \\
&W && = n \text{ MAX}[Link\_distance_i] \\
& && \text{or} \\
& && = 1 + \text{MAX}[Link\_cost_i] \\
&Frequency_i && = \text{number of times node i or link i is used} \\
& && \quad \text{on the K-paths}
\end{aligned}
$$

The second term of the path cost is a penalty for repeated nodes or links. This term is zero for disjoint paths.

The corresponding flow problem for the defined path cost is as follows:

c(u,v) = capacity of link from node u to v
d(u,v) = cost of link from node u to v
f(u,v) = flow from node u to v, no greater than c(u,v)
$\sum$ out-going flow - $\sum$ incoming flow = 0, if not source or destination
= K, if source
= -K, if destination
where K is the number of desired paths

Minimize cost C, where
$C = FC + W \sum_{u,v} MAX[f(u,v) - 1, 0]$
$FC = \sum_{u,v}$ f(u,v)*d(u,v)
or
FC = $\sum_i$ MAX[d(u,v) on path i]

The second term of the cost function, C is a modification to the minimum cost flow problem. This modification produces an extrema for each of the following cases:

1. $W = 0$ and d(u,v) ≥ 0

2. $W \geq 0$ and d(u,v) $= 0$

3. $W \geq 0$ and d(u,v) ≥ 0

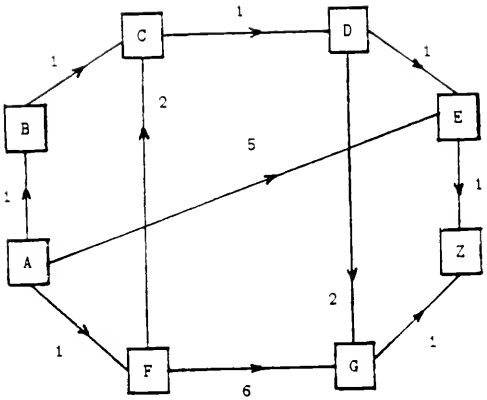
Case 1 can be used for disjoint paths when the capacity is limited to 1. Using an arbitrary capacity for individual links is useful for restricting flow on risky links.

Cases 2 and 3 are used for fewest repeated nodes or links. The weight, W is picked such that the cost of a repeated node or link exceeds the maximum cost of $\mid P \mid$. Hence a minimum total cost set of disjoint paths will be found first, and if additional paths are required, then the fewest possible nodes or links will be repeated.

6.1.2 Discussion of the Approach

Ford [18] and Dantzig [12] provided early solutions to the minimum cost flow problem. Edmonds [15] showed how augmentation along a minimum cost path would produce a minimum cost flow with each iteration. Figure 6.1 shows the steps for producing a minimum cost flow for a network with capacity of 1 for all links. (1) A minimum cost path, P is found from A to Z. (2) Each link on P receives a flow of 1 and the link is reversed. The saturated links have a negative cost because a flow in the reversed link will cancel the existing flow and reduce the total cost of the flow. Adjacent links to the source with flow of 1 cannot be reduced, so are removed. This may not be obvious, but is shown to be true for a minimum cost flow. (3) Step 1 is repeated and another shortest path is found. This produces a flow of 2, and hence 2 link disjoint paths of minimum total cost given by P2.

Dijkstra's shortest path algorithm [14] cannot be used directly on a graph with negative edges. However, it can be used on an equivalent canonic network as shown by Suurballe [35]. The Dijkstra labels, L_i (cost from source to node i) are used to



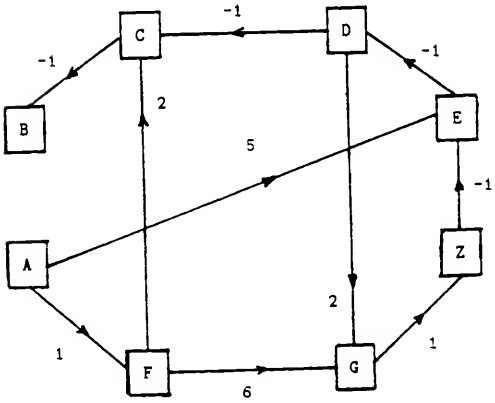


Figure 6.1. Finding A Minimum Cost Flow

transform negative arcs to positive ones:

$$d'_{ij} = (L_i + d_{ij}) - L_j$$

The algorithm for K-paths of minimum total cost with fewest repeated nodes and links is very similar. A minimum cost path is found from source to destination on a transformed graph, G'. The construction procedure for G' provides for arbitrary capacity links and weights for links or nodes with flow greater than 1.

6.1.3 Flow to Graph Constructs

Fewest Repeated Links

Let f[i,j] be the flow from node i to node j, d[i,j] the edge cost, W the repeat link weight, c[i,j] the link capacity, and G be a simple undirected graph. The graph constructs for a flow f[i,j] are shown in figure 6.2.

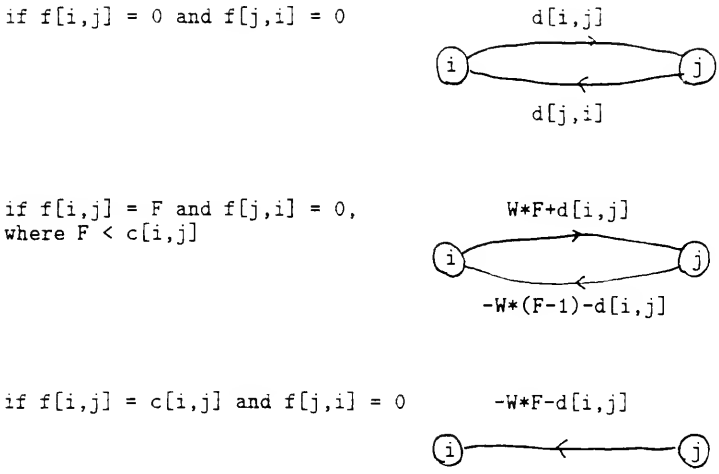


Figure 6.2. Flow to Graph Constructs For Fewest Repeated Links

Explanation of constructs:

1. f[i,j] = 0 and f[j,i] = 0: the flow from i to j or j to i can be increased to 1
2. f[i,j] = F and f[j,i] = 0: the flow from i to j can be increased by 1 or the flow can be decreased by 1
3. f[i,j] = c[i,j] and f[j,i] = 0: the flow can only be decreased by 1

Fewest Repeated Nodes

Let f[i,j] be the flow from node i to node j, d[i,j] the edge cost, c[i,j] the link capacity, W the repeat node weight, Ft[i] the sum of in-coming flow of node i, and G be a simple undirected graph. The graph constructs for a flow f[i,j] are shown in figure 6.3.

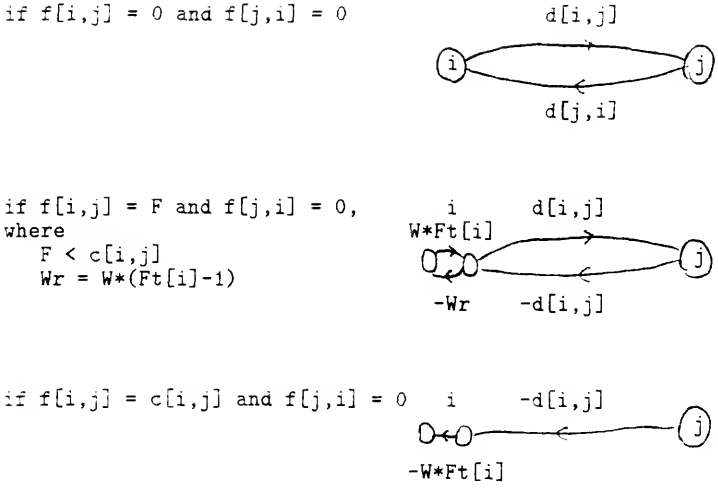


Figure 6.3. Flow to Graph Constructs For Fewest Repeated Nodes

6.1.4 K-Paths Algorithm

Fewest Repeated Links

1. f[i,j] = 0 for all i and j
2. construct G' from G using the graph constructs for f[i,j]
3. remove any arcs on G' which enter the source
4. find the minimum cost path, S on G' from source to destination
   if S does not exist, go to step 7 (K paths do not exist)
5. for each edge on S: if edge from i to j on G' is positive then f[i,j] = f[i,j] + 1; otherwise f[j,i] = f[j,i] - 1
6. if $\sum_j f[source, j] < K$, then go to step 2
7. use a breadth first search on matrix f[i,j] to find the paths from source to destination
   Note: the set of paths is not unique, but all sets have the same total cost

Correctness proof:
The following propositions are made, with proofs or references to proofs given in section 6.1.5.

1. construction of G' from G and f[i,j] provides for all possible minimum cost flows
2. if f[i,j] > 0, then the flow construction has 2 parallel edges from j to i: (1) a negative edge corresponding to a flow reduction (2) a positive edge corresponding to a flow increase; hence the positive edge can be ignored without loss of generality
3. flow augmentation is along the minimum cost path: this guarantees the cost function is minimum for each flow increment
4. the final flow is K if S exists on the last iteration
5. the number of paths is equal to $\sum_j f[source, j]$
6. the number of repeated links is minimum
7. the shortest paths on G' are link simple
8. no flow reduction can occur on links adjacent to the source

A consequence of the minimum total cost set of paths with repeated links is that bi-directional links will not occur. This may seem counter-intuitive, but results from the shortest path always following a link reduction (negative cost) rather than link increment (positive cost).

Fewest Repeated Nodes

The algorithm for fewest repeated nodes is very similar, the only difference is that each node is split into "to" and "from" with an arc of cost equal to the weight connecting "to" and "from". The weight is applied to node i when it is on a minimum cost path S. The node splitting need not be explicit. It can be done implicitly by modifying Dijkstra's shortest path algorithm. The update step becomes:

$$L_j = \text{MIN}[L_j, d_{ij} + L_i + W_j]$$

Correctness proof:
The following propositions are made, with proofs or references to proofs given in section 6.1.5.

1. all propositions made for link disjoint paths
2. the number of repeated nodes is minimum
3. the shortest paths on G' are node simple for all positive links

Maximum Cost Link on Minimum Cost Path

The maximum cost link on the minimum cost path can be found by using Dijkstra's algorithm with the sum function replaced with the maximum function. Hence the update of the label's becomes:

$$L_j = \text{MIN}[L_j, MAX(d_{ij}, L_i)]$$

The proof is as follows: Given: A directed arc graph G(V,E), where V is the set of vertices V = $\{v_1, v_2, \ldots, v_n\}$ and a set of edges E = $\{e_1, e_2, \ldots, e_m\}$ with no self-loops. Assign a cost to each edge and denote the edge from vertex v_i to vertex v_j as u(i,j) with cost c(i,j), where $-\infty \leq c(i,j) \leq \infty$.

Algorithm:

a. Initialize
   Unknown = V - $\{v_A\}$; all vertices are unknown except source
   L[i] = c(A,i), if v_i is adjacent to v_A; the tentative cost
   = ∞, otherwise: of going from vertex v_A to v_i
   Define L[A] = 0

b. Find index s such that L[s] = MIN(L[i]: for all i,
   i an index of the vertex in the set Unknown)

c. Unknown = Unknown - $\{v_s\}$

d. For all vertices v_i adjacent to v_s in the set Unknown:
   L[i] = MIN(L[i],MAX(L[s],c(s,i)))

e. If Unknown is not empty, go to step b

Correctness Proof: Prove by induction that L[i] is the cost of the minimum cost path from v_A to v_i.

basis: At initialization only the origin v_A is not in Unknown and L[A] = 0. Hence the basis is true by definition.

loop invariant: If vertex v_i is not in Unknown, then L[i] is the cost of the minimum cost path from v_A to v_i.

inductive step: Assume that the loop invariant holds on the previous iteration. In the current iteration

1. choose s such that L[s] is the minimum L[i] for all v_i in Unknown

2. Unknown = Unknown - $\{v_s\}$

3. L[x] = MIN(L[x],MAX(L[s],c(s.x))), for all v_x in Unknown

claim i: Any path P_u from v_A to v_s which has vertices from Unknown has cost greater than or equal to L[s]. Assume by way of contradiction that some path P_u has cost C_u <L[s], where P_u contains vertices from Unknown. Then the choice of v_s where L[s] is minimum leads to a contradiction in (1).

claim ii: L[s] is the cost of the minimum cost path from v_A to v_s. We know that there is a path from v_A to v_s using only vertices not in Unknown. By the update procedure L[s] $\leq$ MAX(L[s],c(s.x)), thus L[s] $\leq$ cost of minimum cost path.

6.1.5 Proofs

Lemma 6.1: The 3 flow constructs of figure 6.2 provide for all possible flow augmentations along a minimum cost path on G'.

Proof: Let F_0: f[i,j] = 0 and f[j,i] = 0;

F_F: 0 <f[i,j] < c[i,j] and f[j,i] = 0;

F_c: f[i,j] = c[i,j] and f[j,i] = 0.

Case 1): A link with no flow is represented by F_0. By definition, the flow can only increase. Links from i to j and j to i provide for an increase of flow in either direction. The cost of a flow increase is d[i,j].

Case 2) A link with non-zero flow and not saturated is represented by F_F. The flow in the link can either increase or decrease. A flow increase is feasible with link from i to j. The cost of the increase is W*F+d[i,j]. A flow decrease is feasible with link from j to i. The flow decrease can be represented by two parallel links from j to i: a) a negative cost link, -(F-1)*W-d[i,j] b) a positive cost link, d[i,j]. The flow augmentation is along the minimum cost path. Since -(F-1)*W-d[i,j] < d[i,j], the augmentation path will always follow the negative cost link. Hence the positive cost link from j to i can be neglected without loss of generality.

Case 3) A saturated link is represented by F_c. By definition, the flow can only decrease. A link from j to i of cost -W*F-d[i,j] will reduce the cost function by reducing the flow from i to j by 1 unit.

Lemma 6.2: The construction of G' from G and f[i,j] provides for all minimum cost flows.

Proof: Initially f[i,j] = 0 for all i and j. This state is represented by construct F_0. If a path exists from source to destination, then one or more links will have a flow of 1 after the first flow augmentation. Each link of G will be represented by F_0,F_F, or F_c on G'. By lemma 6.1, all possible flow augmentations can be represented by G'. At any augmentation step $0 \leq i \leq K$, F_0,F_F,F_c represent all feasible flows because the link flow transitions due to augmentation are constrained to be:

$F_0 \rightarrow F_F$ or F_c

$F_F \rightarrow F_0, F_F$, or F_c

$F_c \rightarrow F_F$ or F_0

All flow augmentations are feasible and hence all minimum cost flows can be represented by G' from G and f[i,j].

Lemma 6.3: The cost function C is minimized for each flow increment.

Proof: Theorem 5 in reference [18], p.121 proves a minimum cost augmentation path added to a minimum cost flow of F, produces a minimum cost flow of $F + 1$ for a cost function with $W = 0$. (i. e., no penalty term). Theorem 4 in [15] shows the minimum cost path is a shortest path. It will be shown that the cost function, C' with no penalty term for graph, G' is equivalent to a cost function, C with penalty for graph, G. Let each link on G' with capacity c[u,v] be represented as c[u,v] links of capacity 1. Assign a cost to the i^{th} link as $W * i$+d[u,v]. By references [15,18]

a minimum cost flow can be obtained on G'. Since the flow augmentation always chooses the least cost link, higher cost links can be ignored without loss of generality. Hence the cost, C' of a flow on G' with no penalty term is equivalent to the cost C with link penalty term on G. Proof using the node penalty term is analogus. Split each node into "to" and "from" nodes connected with links of capacity 1 and cost $W * i$ for the i^{th} link. Using the same arguments for the link penalty term proves the cost function C is minimized for each flow increment and this completes the proof.

Lemma 6.4: The number of paths is equal to $\sum_j$ f[source,j].

Proof: Let K' = $\sum_j$ f[source,j]. K' is equal to the number of augmentations since each augmentation increases the flow from the source by 1. Hence there are K' paths from the source to nodes adjacent to the source. By the conservation of flow, the out-going flow must equal the in-coming flow to a node, unless it is the destination. So, if a flow of K_D goes from source to destination with no intermediate nodes, then K'-K_D paths exist from source to destination with one or more intermediate nodes. At any intermediate layer of nodes from the source, there will be a flow of K'-K_D', where K_D' is the flow into the destination node. By the conservation of flow there must be some layer of nodes from the source where K' = K_D' and hence K' paths exist from source to destination.

Lemma 6.5: The final flow is K iff S exists on the last iteration.

Proof: The flow is increased by 1 with each iteration on which S exists. After K iterations, the flow is K. If S exists after K iterations, then S must have existed on the previous K-1 iterations because the algorithm terminates when S does not exist. Hence the final flow is K iff S exists on the last iteration.

Lemma 6.6: The number of repeated nodes or links is minimum.

Proof: Assume some path P' has fewer repeated nodes or links than the minimum cost path, P. The cost of the penalty is f[i,j]*W with W > MAX | P |. Since P < P', a contradiction in the constraint MAX | P |< W exists. Hence the number of repeated nodes or links is minimum.

Lemma 6.7: The minimum cost paths on G' are link simple.

Proof: Assume paths P_1 and P_2 exist on G' between an arbitrary pair of nodes. Suppose the flow augmentation chooses P_1, then $P_1 \leq P_2$. Reversing all links on P_1 results in a cycle of cost $P_2 - P_1 \geq 0$. Hence all cycles produced by the flow augmentation are non-negative. Positive cost loops greater than zero violate the minimum cost path constraint and zero cost loops will not occur in a shortest path algorithm when a node simple path exists. Hence minimum cost paths on G' are link simple.

Corollary 6.1: The flow on a link adjacent to the source cannot be decreased.

Proof: Reduction of flow on a link adjacent to the source implies the source is on a negative cycle. As shown in lemma 6.7, negative cycles do not exist on G'. Hence the flow on adjacent links cannot be decreased.

6.1.6 Conclusions

An algorithm is presented whichs finds K-paths of minimum total cost with fewest repeated nodes or links. The running time is the same as that for finding a shortest path on a graph. The algorithm can be used for multiple path routing for increased reliability. Two different metrics are given: (1) propagation delay (2) retargeting frequency. The latter metric is useful for rapidly changing topologies such as the proposed SDI communications network.

The solution to the minimum cost flow problem with penalty term is used. A graph to capacity flow transformation is given.

6.2 A Quick Algorithm for Disjoint Paths

Suurballe [35] has given an O(K $n^2 \log(n)$) algorithm for node disjoint paths of minimum total distance between a single source and all destinations. Another important metric is remaining path time, where link cost is the reciprocal of remaining link time. For low altitude satellite networks, remaining link time may be only minutes. Hence to reduce the number of path changes, the minimum sum of the remaining path time reciprocals can be used.

Let G be an undirected graph containing m links (edges) of non-negative cost and n nodes (vertices). The cost of a path is defined to be the maximum cost link on the minimum cost path. An algorithm is presented which finds a maximal number of disjoint paths of minimum total cost between any pair of nodes on G. The running time of the algorithm for a single source to all $n-1$ destinations is O(K nm), where K is the number of disjoint paths.

6.2.1 Discussion of the Approach

The maximum cost link on the minimum cost path can be found by using Dijkstra's algorithm with the sum function replaced with the maximum function. Hence the update of the label's becomes:

$$L_j = \text{MIN}[L_j, MAX(d_{ij}, L_i)]$$

The path cost of the maximum cost link on the minimum cost path can only be from the set of link costs rather than the universal set. Using this fact will provide

for a faster algorithm than Dijktra's shortest path algorithm. The speed up comes in finding the minimum cost label, which can be done in constant time rather than $\log n$ time.

The minimum cost label can be found in constant time for the maximum cost edge on the minimum cost path by using a link cost transformation. The transformation uses the index of the sorted list of link costs. If two links have the same cost, then they have the same index. Hence the range of the link costs is from 1 to the number of edges.

The algorithm uses the flow to graph constructs given in 6.1, however the link flow is constrained to be 0 or 1. The following sections give the algorithm and proofs.

6.2.2 Disjoint Path Algorithms

Link Disjoint

1. f[i][j] = 0 for all i and j (flow from i to j)
2. generate adjacent node list for each node i.
   AdjNodes[i][k]: i = 1,..,n and k = 1...,DegreeOfI
3. sort links into increasing order, place in SortedLinks[i]
4. assign link numbers:
   count = 1; LastLink = -1
   for i = 1,...,NumberOfLinks
   DO BEGIN
   LinkNumber[i] = count
   if (SortedLinks[i] NE LastLink) THEN count = count + 1
   LastLink = SortedLinks[i]
   END i
5. con1[i][j] = LinkNumber[], if edge on input graph (undirected) G exists, otherwise con1[i][j] = MAXINT (edge does not exist)
6. for each destination node j = 1,...,n
   BEGIN
7. for each path m = 1,...,NumberDisjointPaths
   BEGIN
8. use MaxMin algorithm to find a path, S from source to j (use con1[i][j] and AdjNodes[i][k])

9. for each edge on S between i and j: if (con1[i][j] > 0)
   THEN BEGIN con1[i][j] = con1[i][j] + W; f[i][j] = 1; con1[j][i] = -con1[j][i]; END
   if (con1[i][j] < 0)
   THEN BEGIN f[j][i] = 0; con1[j][i] = con1[j][i] - W; con1[i][j] = -con1[i][j]; END
10. END m
11. use breadth first search on f[i][j] to find the paths from source to destination (use AdjNodes[i][k])
    Note: f[i][j] and con1[i][j] are restored to initial state with:
    if (f[i][j] = 1)
    THEN BEGIN f[i][j] = 0; con1[i][j] = con1[i][j] - W; con1[j][i] = -con1[j][i]; END
12. output disjoint paths from source to j
13. END j

Correctness proof:

The propositions made in section 6.1.4 for the K-paths algorithm apply here. The proofs or references to proofs are given in section 6.2.3.

Node Disjoint

The algorithm for node disjoint paths is very similar, the only difference is that each node is split into "to" and "from" with an arc of cost equal to the weight connecting "to" and "from". The weight is applied to node i when it is on a minimum cost path S. The node splitting need not be explicit. It can be done implicitly by modifying the maximum cost link on the minimum cost path algorithm. The update step becomes:

$$L_j = \text{MIN}[L_j, MAX(d_{ij}, L_i) + W_j]$$

Correctness proof:

The following propositions are made, with proofs or references to proofs given in section 6.2.3.

- all propositions made for link disjoint paths
- the shortest paths on G' are node simple for all positive links

Maximum Cost Link on Minimum Cost Path

Determine the cost and path of the maxminum cost link on the minimum cost path from source to destination.

The algorithm is:

1. initialize link cost bins, best distance labels, and path tree:
   for i = 1....NumberOfEdges DO Bin[i] = 0
   for i = 1,...n DO BEGIN
   L[i] = con1[source][i]; tree[i] = source; Bin[L[i]] = i; END i

2. for i = 1 to NumberOfEdges
   BEGIN if (Bin[i] > 0)
   BEGIN
   for all nodes j at i:
   for all nodes k adjacent to j:
   if (L[k] > MAX(L[j],con1[j][k])) THEN BEGIN
   tree[k] = j; L[k] = MAX(L[j],con1[j][k]); add k to Bin[L[k]]; END if

3. END i,j,k

Correctness Proof:

Lemma 6.8: For each pair of link costs d[i,j] and d[p,q], there is a relation $R_{ijpq} = \{<, =, >\}$ which is identical to R'_{ijpq} for $LinkNumber_{ij}$ and $LinkNumber_{pq}$.

Proof: Assume d[i,j] R_{ijpq} d[p,q] and $LinkNumber_{ij}$ R'_{ijpq} $LinkNumber_{pq}$. $R_{ijpq} = R'_{ijpq}$ because the sorting and assignment rule preserve $\{<, =, >\}$. Conversely, assume $LinkNumber_{ij}$ R'_{ijpq} $LinkNumber_{pq}$ and d[i,j] R_{ijpq} d[p,q]. $R_{ijpq} = R'_{ijpq}$ because the elements d[x,y] are sorted for all x,y and the assignment rule preserves $\{<, =, >\}$.

Lemma 6.9: Compression of link values d[i,j] into $LinkNumber_{ij}$ does not change

the minimum cost path P, where path cost is equal to the maximum cost link.

Proof: The best distance labels, L_j are determined by the rule:

$$L_j = \text{MIN}[L_j, \text{MAX}(d_{ij}, L_i)]$$

Hence L_j is determined only by the maximum and minimum operators. By lemma 6.8, $R_{ijpq} = R'_{ijpq}$, so the operators choose the same links. Hence compression of link values does not change the minimum cost path.

6.2.3 Proofs

Lemmas 6.1 through 6.7 prove the general algorithm for fewest repeated nodes or links. No repeated nodes or links is a special case of the general algorithm.

6.2.4 Conclusions

An algorithm is presented whichs finds disjoint paths of minimum total cost. The running time is the same as that for flow augmentation. The algorithm can be used for multiple path routing for increased reliability.

CHAPTER 7
EVENT DRIVEN SIMULATION

An event driven simulation can run on one or more processors and can be implemented at the packet level or M/M/1 queueing equation level. The SDI communications network was simulated using both approaches. The novelty of the packet level simulation is that it runs on multiple processors with a dynamically reconfigurable network [31,34,38]. Each satellite was modeled using a DSP32 processor. The reconfigurable network reflected the link assignment. The novelty of the M/M/1 queueing equation level simulation was that it would handle step changes in the traffic matrix.

The importance of using these two novel approaches is that a conventional packet simulation on a uniprocessor is not feasible for large networks. The M/M/1 queueing equation level offers a speed-up proportional to the packet transmission rate. A linear speed-up in the simulation can be made by using a processor for each satellite, and connecting all processors with high speed communication links.

Section 7.1 discusses the packet level simulation using multiple procesors and section 7.2 discusses the M/M/1 queueing equation level simulation.

7.1 Distributed Packet Level Simulation

A simulation of a rapidly changing satellite network with laser cross links uses an AT&T DSP32 processor for each satellite. The processors are connected via a reconfigurable network which reflects the time varying topology. One or more laser channels can be optionally inserted between the processors. A combined discrete event and discrete time simulation is used for packet transfer between satellites [6,39].

Each packet contains time so that the simulation proceeds at the rate of the slowest processor. Since the minimum delay between satellites is several milliseconds and packet transfer rates of at least 10 per millisecond are used, each satellite can process packets as discrete events during the minimum delay window.

7.1.1 Objectives and Measurements

The simulation provides a test of the algorithms used for link assignment, routing, link failure detection, link reconfiguration, and data link protocols. Many different algorithms have been developed and performance can be measured using various topologies. Performance measures for link assignment include connectivity, retargeting frequency, and propagation delay. Measures for routing include end-to-end delay, rerouting frequency, and number of common satellites on multiple paths between origin and destination. Minimizing the number of common satellites on multiple paths is an important consideration for survivable military communications.

7.1.2 Event Processing

The simulation proceeds based on the slowest processor (satellite). Each processor has a counter which marks discrete time. The counter is included in each packet transmitted. When a processor receives a packet with a smaller count, that processor sets its counter to the smaller value.

The synchronization of clocks is not critical since each packet transmitted has a minimum propagation delay of several milliseconds. Hence each processor can run a discrete event simulation during the interval from the processors clock to processor clock plus minimum propagation delay.

The clock synchronization assumes time packets will be transmitted if no other packets are available to be sent. This adds overhead to lightly loaded links, but not to heavily loaded links. Time packets may be used in the real system as well

because link establishment is done in advance and synchronized clocks are required to eliminate waiting caused by a satellite with a slower clock.

7.1.3 Hardware Architecture

A multiprocessor simulation of a high data rate communications network requires that all processors be interconnected via some high speed communications network. This is necessary because packet transfers must be passed between the processors and all packets must be processed in proper time sequence. For discrete event simulation only the packet headers need be passed between the processors. However, if 100 packets/msec links are to be simulated, transfering only packet headers can be a bottleneck in the simulation.

The high speed parallel and serial ports of the AT&T DSP32 are connected to DMA channels, which make it well suited as a processing element in a distributed architecture. One architecture which is being used on a board manufactured by *DSP Applications, Inc.* uses one slow speed DSP32 as a controller for four high speed DSP32C processors. The controller has the parallel ports of the DSP32C processors connected to its 32 bit wide data bus. Since the parallel ports are either 8 or 16 bits wide, the DSP32 controller can read or write two or four processors in a single instruction cycle. The serial ports on the controllers can be connected in a ring to form a 2-connected network. In addition, the serial ports on the DSP32C processors can be connected in a variety of ways to form a 3-connected network. The serial port interconnection need not be static, 64X64 crossbar chips are available from Texas Instruments which could be used to dynamically configure the DSP32C serial ports. This architecture offers an improvement over the AT&T ASPEN machine which uses the serial ports connected into a binary tree. In a binary tree, nodes near the root become a bottleneck. This can be avoided by using a ring, at the expense of a longer

worst case path between processors. But this worst case path can be reduced by connecting DSP32C processors as cords on the ring (using serial ports).

Use of a controller for four processors offers two kinds of satellite network simulations: 1) each DSP32C can be modeled as a satellite, 2) each DSP32C can be modeled as a transmit/receive processor, so the satellite is modeled by the controller and four DSP32C processors. The latter configuration offers four times the data rate, but fewer nodes. This type of simulation is useful for real-time demonstrations.

An important advantage of the controller, four DSP32C processor architecture is the modularity. Each DSP32C with 128 Kbytes of memory and 25 MFLOPS of processing capability can be contained on a 4" by 4" plugable card. A single board for the IBM PC could contain a DSP32 controller and four DSP32C plug-in cards. This would offer 100 MFLOPS of processing power. Multiple controller cards could be used and all DSP32C processors would be fully connected via the controllers. Hence any satellite topology could be accommodated via the parallel ports or the serial ports.

The parallel port on the DSP32 controller is used to communicate with a host for antenna direction control. The host controls the network configuration based on the antenna direction commands received from each DSP32 controller.

7.1.4 Software Architecture

The software in each node is identical and executes as a single process. A node could be either a single DSP32C or an IBM PC host with DSP32 controller and four DSP32C processors. The software uses an object oriented approach. The objects are satellites and packets. The following tables give the attributes of each.

SATELLITE

Attribute	Description

CurrentAntenna[4]	list of satellites pointed at
FutureAntenna[4]	list of satellites to be pointed at
FutureAntTime[4]	time antennas will be pointed
CurPathPointer[4]	pointer to list of satellites on path
FurPathPointer[4]	pointer to list of satellites on future path
Position[3]	x,y,z coordinates of position
NextPacketTime	time next packet will be generated

PACKET

Attribute	Description
Source	source satellite number
Destination	destination satellite number
PathNumber	path number for disjoint paths
Stime	start time
Atime	arrival time
DataField	packet data

Each satellite contains a description of all other satellites in the network. The descriptions are updated based on packet exchange and predicted satellite positions. It is assumed that each satellite has four antennas (the number of processors time multiplexed). Hence the dimension of four on the satellite attributes. The four path pointers are for multiple paths. For example, the satellite j description by satellite i would have the first satellite on the path as i and the last as j.

Link assignment and routing can be done in advance since orbital mechanics are used to predict future satellite positions. Hence no transmission time is lost due to the overhead of executing the algorithms.

Each packet has a path number and an optional data field. The path number is used to uniquely define the packet route when node disjoint routing is used. Other routing algorithms use only the destination and ignore path number. The data field can be null for information packets. This reduces the simulation communication between processors. The simulation statistics reflect the transmission of a full data field by adjusting arrival times (Atime) to be propagation delay plus transmission time of a full packet.

The transmission queues (propagation) are combined with the satellite antenna queues for ease in processing. The Atime is the time the packet will arrive at the satellite. Hence, the packet stays in the queue until the simulation clock is greater than Atime. The queues are sorted based on Atime. For packets being generated, the Stime and Atime are the same. When a packet reaches its destination, the delay statistics are computed and the packet disappears.

Routing and link control are passed via the data field. For most packets this field is null, but for packets where link change information is needed or protocol testing is desired, the field is filled in.

The event handler provides two services (1) it checks all in-coming packets for routing or link control information and passes them to the appropriate control routines, (2) passes all other packets to the routing routine for satellite queue determination or if it has reached its destination, it passes then to the packet delay statistics routine.

7.1.5 Software Module Description

The satellite simulation program is written in C-language. The modules are include files, which facilitate algorithm testing. A common set of procedure calls are used for each module. Hence, if a different link assignment algorithm is to be tested, only the include file name need be changed. The following table provides the functions of the current modules.

Module Name	Functions
connect.h	disjoint paths, graph connectivity
topology.h	3-D rotation, satellite visibility, orbits
kpaths.h	multiple path routing algorithms
neighbor.h	nearest neighbor link assignment
packet.h	packet generation and queueing
schedule.h	event handling, simulation control

7.1.6 Conclusions

A DSP32 multiprocessor architecture is described which will accommodate high speed simulation of large networks. The processors are 3-connected with a dynamic topology. The topology can be made to match a rapidly changing satellite network topology. The advantages of the multiprocessor architecture include: 1) modularity, 2) high speed communication, 3) nodes can be modeled as a single processor or four processors, 4) dynamic reconfigurable network, 5) all processors are fully connected via DSP32 controllers.

The software architecture is also modular, using include files to implement the network communications software. A common calling convention allows link assignment

and routing algorithms to be swapped without affecting the rest of the simulation program.

7.2 M/M/1 Queueing Level Simulator

The M/M/1 notation means that packet generation is done using an exponential distribution. This should be valid for target report generation which is expected to account for 80% of the SDI traffic. Other assumptions of M/M/1 include unbounded buffers and first-come-first-serve scheduling of the queues. The latter restriction does not mean that pre-emption cannot be used for priority packets. If priority packets are few in number, then the queueing delay can be neglected, since it should be small compared to the propagation delay.

The principal advantages of M/M/1 for the SDI simulation are:

1. analytical solutions can be used to reduce simulation time
2. the simulation results reflect the true average characteristics of the network
3. transient responses can be studied

The disadvantages of M/M/1 include:

1. worst case conditions may be hard to represent
2. M/M/1 assumptions may not be valid

Because of link and routing changes, an M/M/1 simulation would not be appropriate for SDI. However, a mixture of M/M/1 and packet level could certainly be valid. The following sections describe the equations and algorithms for computing the packet end-to-end delay for a transient response. It is assumed the SDI traffic will be a series of traffic steps.

7.2.1 Total Delay

The total elapsed time between when a packet is sent and when a packet is received is the total delay. This delay contains propagation delay and queueing delay. The propagation delay is constant if the topology does not change and is simply the ratio of distance and velocity of light in a vacuum. The queueing delay varies and has a minimum time equal to the time it takes to transmit one packet in a store and forward network. In a relay node, the queueing time can be zero because the packet is not stored before it is re-transmitted. For high data rate satellite networks, the transmission time is microseconds while the propagation time is milliseconds, so the packet store time can be neglected (swamped by propagation time).

For the M/M/1 equation level simulator an algorithm is needed to first compute the link flows and then add the propagation delay and queueing delay. The queueing delay is given by:

$$queue\_delay = \frac{1}{link\_capacity - link\_flow} \tag{7.1}$$

The algorithm for finding the total network delay is:

```
TrafficFlow[i,j] = packets/sec from satellite i to satellite j

ProbabilityOfUse[i,j,k] = probability of using path k from
                          satellite i to satellite j
                          /* k = 1,2,3,4 */

for i = 1 to Np*Ns
   for j = (i+1) to Np*Ns
      {
       generate 4 node disjoint paths from i to j
       for k = 1 to 4   /* node disjoint paths 1,2,3,4 */
            {
             PropagationDistance = sum of D values from
                                   closed formula
             P = ProbabilityOfUse[i,j,k]
             PropagationDelay[i,j] = PropagationDelay[i,j] +
                          P*PropagationDistance/SpeedOfLight
             for nodes = 1 to nodes_on_path_k - 1
```

```
                {
                 p = from_node
                 q = to_node
                 Flow[p,q] + Flow[p,q] + P*TrafficFlow[i,j]
                 Flow[q,p] = Flow[q,p] + P*TrafficFlow[j,i]
                }
            }
        }
C = link_capacity      /* link_capacity > Max(Flow[i,j]) */
for i = 1 to Np*Ns
  for j = (i+1) to Np*Ns
     {
      TotalDelay[i,j] = PropagationDelay[i,j]
      TotalDelay[j,i] = PropagationDelay[i,j]
      generate 4 node disjoint paths from i to j
      P = ProbabilityOfUse[i,j,k]
      for k = 1 to 4    /* node disjoint paths 1,2,3,4 */
         for nodes = 1 to nodes_on_path_k - 1
             {
              p = from_node
              q = to_node
              TotalDelay[i,j] = TotalDelay[i,j] +
                                      P/(C-Flow[p,q])
              TotalDelay[j,i] = TotalDelay[j,i] +
              P/(C-Flow[q,p])
             }
     }
```

7.2.2 Transient Response

The transient response for the packet level simulator only requires monitoring the packet delays and computing a running average over some time interval. A convenient interval is one millisecond. This interval is short compared to link propagation delays (> 50 milliseconds), but long enough to average out the delay flucuations caused by exponential arrival times.

The transient response for the M/M/1 queueing equation level simulator uses the following algorithm:

1. compute the steady-state total delay at time T_i for an Np.Ns network with traffic matrix TM(T_i) and routing probability matrix RP(T_i)

2. compute the steady-state total delay at time T_{i+1} for an Np.Ns network with traffic matrix TM(T_{i+1}) and routing probability matrix RP(T_{i+1})

3. find propagation times from each source node to each node along the path to the destination node for all 2*Np*Ns*(Np*Ns-1) paths

4. store the from node, to node, and corresponding propagation delay plus linear approximation to queue delay

$$
\begin{aligned}
QueDelay(T_i) &= \frac{1}{LinkCap - LinkFlow(T_i)} \\
QueDelay(T_{i+1}) &= \frac{1}{LinkCap - LinkFlow(T_{i+1})} \\
DT &= (T_{i+1}) - (T_i) \\
QueueDelay(t) &= QueDelay(T_i) + \frac{t - T_i}{DT}(QueDelay(T_{i+1}) - QueDelay(T_i))
\end{aligned}
$$

5. order the times in a list, $T_{S1}, T_{S2}, \ldots, T_{Sn}$

6. for T = $T_{S1}, T_{S2}, \ldots, T_{Sn}$

   for all nodes in the list at time T

   $Fij(T_k) = Fij(T_k) + \triangle TMsdRPsd * (T_k - T_i)/DT$

where Fij(T_k) is the steady-state flow from i to j at time T_k

$$\triangle TMsdRPsd = TMsd(T_{i+1})RPsd(T_{i+1}) - TMsd(T_i) * RPsd(T_i)$$

TMsd(T_i) is the source to destination traffic flow at time T_i

7.2.3 Conclusions

An analytical method for computing the step response of a network is presented which gives the average delay as a function of time. This approach is valid for

networks where propagation delay is long compared to queueing delay. The principal advantages of this approach include: 1) average packet delay can be computed for large networks, 2) true M/M/1 average delays are computed, 3) pre-emption can be accommodated for priority packets.

CHAPTER 8
CONCLUSIONS

This dissertation considers the SDI communications network as a system. Metrics for the performance of the system were given and solutions were derived which optimized the metrics. Section 8.1 gives the significant results of the optimization. The close association of topology, link assignment, routing, and performance metrics are given in section 8.2. Finally, future work that could be done is given in 8.3.

8.1 Significant Results

Topology optimization was possible using the closed formulas derived for the N_p, N_s model. The metrics were propagation delay and connectivity. Algorithms were developed for more general topologies.

Link assignment was optimized using the N_p, N_s mesh. The connectivity was optimal for N_p, N_s topologies with 4 antennas and general topologies were near optimal. An analytical solution was derived for the propagation delay of the N_p, N_s mesh.

Routing algorithms were developed which have a primary objective of minimum total cost and a secondary objective of fewest repeated nodes or links. The algorithm for K-paths of minimum total cost with fewest repeated nodes or links is a generalization of Suurballe's node disjoint algorithm. The generalization makes use of a modified minimum cost flow problem which has a penalty term for multiple uses of a node or link. The generalization also includes the linear metric of distance and the non-linear metric of maximum cost link on the minimum cost path. The latter metric is important when the overhead of retargeting is appreciable. This metric also offers

an asymptotically faster algorithm for the single source to all destinations routing problem.

8.2 Metrics

The source to destination delay of packets consists of queueing delay and propagation delay. Using a single metric such as propagation delay for optimizing link assignment can result in a small improvement in delay at the expense of reduced connectivity and increased hops per path. These latter two metrics are important for multiple path routing and congestion. Simulation of the N_p, N_s mesh link assignment has shown that shortest paths using distance as a metric may have several more nodes than a shortest path using hops as a metric. In addition, comparision of hop counts for shortest distance paths using 4 antennas and unlimited antennas shows the longer paths with several more nodes. To minimize congestion, paths should have the fewest possible nodes. The N_p, N_s mesh link assignment may not be the best possible compromise between propagation delay, connectivity, and hops per path. A heuristic such as N_p, N_s mesh could be improved by using a penalty for each link used on a path. This might also reduce the number of repeated nodes or links in the K-paths routing algorithms.

8.3 Association of Algorithms and Performance

The SDI communications network can be considered at the system level with inputs and outputs. The inputs can be constrained to be injected packets and link disturbances; while the ouputs are delivered packets. The topology, link assignment, and routing algorithms are part of the system and effect the end-to-end delay of the packets.

The association between topology, link assignment, and routing algorithms in this dissertation is principally connectivity. The number of disjoint paths is limited by the

link assignment, which in turn is limited by the visibility matrix of the topology. The performance measures are metrics to evaluate how well the topology, link assignment, and routing algorithms work individually and as a whole system. Simulation provides qualitative performance results for the algorithms using general topologies.

8.4 Extensions of the Research

Metric

The link assignment and routing algorithms use a single metric for the primary or secondary objective. Some combination of metrics could be used.

Using Feedback From the Routing Algorithm

Algorithms have been given for link assignment and for routing, but the algorithms work independently with the common objective of multiple paths of fewest common nodes or links. An algorithm with feedback from routing to topology could produce fewer common nodes or links. If the topology is considered fixed (such as a deployed SDI system), then the feedback could be from routing to link assignment.

Neural networks have been used to solve optimization problems with feedback. Perhaps n^2 perceptrons would be needed to model n satellites, but the perceptrons can be implemented on a digital computer. The ideal digital computer would have a processor for each perceptron. Such an implementation has become feasible with cheaper processors.

APPENDIX A
NODE DISJOINT PATHS

Tables A.2 and A.4 contain the node disjoint paths for the mesh link assignment using topologies of $Np = 2$, $Ns = 3$ and $Ns = 4$ respectively. The satellite numbers are assigned according to the following equation:

$Satellite\_Number = 1 + Ns * i + j$, where $i = 0, \ldots, Np - 1$ and $j = 0, \ldots, Ns - 1$

Table A.6 has the time invariant node disjoint paths for $Np = 2$, $Ns = 3$.

Table A.1. Adjacency Matrix for $Np = 2$, $Ns = 3$

connectivity from i to j						
j \ i	1	2	3	4	5	6
1	0	1	1	1	0	1
2	1	0	1	1	1	0
3	1	1	0	0	1	1
4	1	1	0	0	1	1
5	0	1	1	1	0	1
6	1	0	1	1	1	0

Table A.2. Node Disjoint Paths for $Np = 2$, $Ns = 3$

node disjoint paths								
1	6	5	2		3	6	4	
1	4	2			3	5	4	
1	3	2			3	2	4	
1	2				3	1	4	
1	6	3			3	6	5	
1	4	5	3		3	5		
1	3				3	2	5	
1	2	3			3	1	4	5
1	6	4			3	6		
1	4				3	5	6	
1	3	5	4		3	2	4	6
1	2	4			3	1	6	
1	6	5			4	6	5	
1	4	5			4	5		
1	3	5			4	2	5	
1	2	5			4	1	3	5
1	6				4	6		
1	4	6			4	5	6	
1	3	6			4	2	3	6
1	2	5	6		4	1	6	
2	5	3			5	6		
2	4	6	3		5	4	6	
2	3				5	3	6	
2	1	3			5	2	1	6
2	5	4						
2	4							
2	3	6	4					
2	1	4						
2	5							
2	4	5						
2	3	5						
2	1	6	5					
2	5	6						
2	4	6						
2	3	6						
2	1	6						

Table A.3. Adjacency Matrix for $Np = 2$, $Ns = 4$

connectivity from i to j								
j \ i	1	2	3	4	5	6	7	8
1	0	1	0	1	1	0	0	1
2	1	0	1	0	0	1	1	0
3	0	1	0	1	0	1	1	0
4	1	0	1	0	1	0	0	1
5	1	0	0	1	0	1	0	1
6	0	1	1	0	1	0	1	0
7	0	1	1	0	0	1	0	1
8	1	0	0	1	1	0	1	0

Table A.4. Node Disjoint Paths for $Np = 2$, $Ns = 4$

node disjoint paths																
1	8	7	2		3	7	8	4		6	7					
1	5	6	2		3	6	5	4		6	5	8	7			
1	4	3	2		3	4				6	3	7				
1	2				3	2	1	4		6	2	7				
1	8	7	3		3	7	8	5		6	7	8				
1	5	6	3		3	6	5			6	5	8				
1	4	3			3	4	5			6	3	4	8			
1	2	3			3	2	1	5		6	2	1	8			
1	8	4			3	6				7	8					
1	5	4			3	7	6			7	6	5	8			
1	4				3	4	5	6		7	3	4	8			
1	2	3	4		3	2	6			7	2	1	8			
1	8	5			3	7										
1	5				3	6	7									
1	4	5			3	4	8	7								
1	2	6	5		3	2	7									
1	8	7	6		3	7	8			5	8	7				
1	5	6			3	6	5	8		5	6	7				
1	4	3	6		3	4	8			5	4	3	7			
1	2	6			3	2	1	8		5	1	2	7			
1	8	7			4	8	5			5	8					
1	5	6	7		4	5				5	6	7	8			
1	4	3	7		4	3	6	5		5	4	8				
1	2	7			4	1	5			5	1	8				
1	8				4	8	7	6								
1	5	8			4	5	6									
1	4	8			4	3	6									
1	2	7	8		4	1	2	6								
2	7	3			4	8	7			2	7	6				
2	6	3			4	5	6	7		2	6					
2	3				4	3	7			2	3	6				
2	1	4	3		4	1	2	7		2	1	5	6			
2	7	8	4		4	8				2	7					
2	6	5	4		4	5	8			2	6	7				
2	3	4			4	3	7	8		2	3	7				
2	1	4			4	1	8			2	1	8	7			
2	7	8	5		5	8	7	6		2	7	8				
2	6	5			5	6				2	6	5	8			
2	3	4	5		5	4	3	6		2	3	4	8			
2	1	5			5	1	2	6		2	1	8				

Table A.5. Adjacency Matrix for $Np = 2$, $Ns = 3$

connectivity from i to j						
j \ i	1	2	3	4	5	6
1	0	1	1	1	0	0
2	1	0	1	0	1	0
3	1	1	0	0	0	1
4	1	0	0	0	1	1
5	0	1	0	1	0	1
6	0	0	1	1	1	0

Table A.6. Node Disjoint Paths for $Np = 2$, $Ns = 3$

node disjoint paths									
1	4	5	2		3	6	4		
1	3	2			3	2	5	4	
1	2				3	1	4		
1	4	6	3		3	6	5		
1	3				3	1	4	5	
1	2	3			3	2	5		
1	4				3	6			
1	3	6	4		3	2	5	6	
1	2	5	4		3	1	4	6	
1	4	5			4	6	5		
1	3	6	5		4	5			
1	2	5			4	1	2	5	
1	4	6			4	6			
1	3	6			4	1	3	6	
1	2	5	6		4	5	6		
2	5	6	3		5	6			
2	3				5	4	6		
2	1	3			5	2	3	6	
2	5	4							
2	3	6	4						
2	1	4							
2	5								
2	3	6	5						
2	1	4	5						
2	5	6							
2	3	6							
2	1	4	6						

APPENDIX B
DUAL DSP32 BOARD

The Dual DSP32 board was designed for digital signal processing and general purpose multiprocessor applications. Its use in the distributed simulation for the SDI communications network is described in section 7.1.

B.1 Hardware Operation

PC Interface

Data flows between the PC and DSP32 via bus transceiver IC2, 74LS245. The direction of the PC bus is determined by $\overline{IOR}$. A low on $\overline{IOR}$ or $\overline{IOW}$ is used to gate data from/to the PC bus. The base port address is determined by comparator IC3, 74LS688. The selection of DSP32A, DSP32B, or timer is performed by demultiplexer IC5, 74LS139.

The DSP32 has an 8 bit data bus to the PC. The data lines are PDB0-PDB7. There are 10 addresses on the DSP32, the last six addresses are replication of addresses 8,9. The address lines are PAB0, PAB1, PAB2, and PACK.

PINT from the DSP32 goes high for an interrupt to the PC if the appropriate switch is on.

Serial Port

The DSP32 serial port is terminated in a DIP socket. For full-duplex operation between two DSP32 processors, the following signal lines should be connected:

Processor A		Processor B	
Pin	Signal	Pin	Signal

14	DO	12	DI
13	DI	15	DO
9	OCK	9	ICK
10	ICK	10	OCK
12	OLD	5	ILD
11	ILD	3	OLD

External Memory

The DSP32 memory mode select signals are MMD0 and MMD1. There are 4 modes, each mode provides a different mapping of the DSP32 internal memory to absolute addresses. The board is wired for mode 2, hence addresses 0xE000-0xFFFF are from the DSP32 and not from the external memory.

The DSP32 external memory address lines are AB00-AB13. PDB0 at address 0xDFFF is used to control the selection of the two. 56 Kbyte pages. A write into 0xDFFF, writes into external memory as well as the page bank selection flip-flop. The flip-flop cannot be read, so a read from this location returns the memory value rather than the contents of the flip-flop.

B.2 Software Operation

DSP32 Interface to PC

The DSP32 has 6 registers called PIO (parallel I/O) which exist on the PC bus as 10, 8 bit ports [3,4]. The PC port address is relative, the absolute address is the relative address plus base address set by DIP switches 1-5.

The following table gives a brief description of the PIO registers.

Port	Register Name	Description
0x0	PARa(l)	DSP32A memory address, 16 bits
0x1	PARa(h)	
0x2	PDRa(l)	DSP32A memory data register, 16 bits
0x3	PDRa(h)	
0x4	EMRa(l)	DSP32A error mask register, 16 bits
0x5	EMRa(h)	
0x6	ESRa	DSP32A error status register
0x7	PCRa	DSP32A PIO control register
0x8	PIRa(l)	DSP32A interrupt vector, 16 bits
0x9	PIRa(h)	
0x10	PARb(l)	DSP32B memory address, 16 bits
0x11	PARb(h)	
0x12	PDRb(l)	DSP32B memory data register, 16 bits
0x13	PDRb(h)	
0x14	EMRb(l)	DSP32B error mask register, 16 bits
0x15	EMRb(h)	
0x16	ESRb	DSP32B error status register
0x17	PCRb	DSP32B PIO control register
0x18	PIRb(l)	DSP32B interrupt vector, 16 bits
0x19	PIRb(h)	

The PAR register is used to select the DSP32 memory address. This register can be selected to autoincrement after each read or write by setting PDB4 to 1 in the PCR. Since 16 bit transfers are always used, the PAR register does not use the least significant bit. When PAR(l) is read, the least significant bit is always 1.

The PCR is used to control the DSP32. The following table gives a brief description of each bit.

Bit	Mnemonic	Function
0	Reset	=0 halt DSP32, =1 run DSP32
1	Intmode	=0 8 bit PIR, =1 16 bit PIR
2	ENI	=0 disable PIR interrupt, =1 enable PIR
3	DMA	=0 DMA disabled on PIO, =1 enabled
4	AUTO	=0 no autoincrement of PAR, =1 autoincrement
5	PDF	=1 when PDR is written, =0 when PDR is read
6	PIF	=1 when PIR is written, =0 when PIR is read

7	REF	=0 refresh disabled, =1 refresh enabled

The PDR register is used to hold the data to/from the DSP32. This 16 bit latch is loaded by the DMA controller or by program control using a move instruction.

The EMR masks conditions in the ESR which cause interrupts to the PC. These conditions include memory parity error, DAU error, addressing error, loss of sanity on the serial data stream, and loss of sync on the serial data stream.

Timer

The programmable timer, 8254 has 4 ports with external count synchronization signal. The timer ports and synchronization signal have relative addresses as shown in the following table.

Port	Register Name	Description
0xC	Count 0	16 bit count of counter 0
0xD	Count 1	16 bit count of counter 1
0xE	Count 2	16 bit count of counter 2
0xF	MODE	control count mode of counters 0-3
0x1C	GATE	enable counters 0-3

The sequence for starting a counter is: (1) the MODE register is written to select a counter and mode (2) two writes (low byte, then high byte) into the COUNT register (3) a write to GATE.

The counting modes include interrupt on terminal count, programmable one-shot, rate generator, square wave generator, software triggered strobe, and hardware triggered strobe.

B.3 Satellite Node Software

The node software will consist of data communications drivers in the DSP32 processors and all the other network functions will be performed in the 80386-PC. Example program segments are included to show the simplicity of accessing the DSP32 from the PC and also how to program DSP32 to DSP32 serial communications.

C-langauge programs for the link assignment and routing algorithms exist, but are too long to be included here. A description of a distributed version of the programs is given in section 3.3.1.

Example Programs

Reading/Writing DSP32 Memory

The following TurboC [1] program segment writes the integers 0-9 into memory locations 0x1000-0x1009 and then reads the integers back.

```
outportb(PCR,0x18); /* halt DSP32 and enable DMA autoincrement */
outport(PAR,0x1000);/* set DMA address */
for (i = 0; i < 10; i++)
    outport(PDR,i); /* write integer into memory */
outportb(PCR,0x18); /* halt DSP32 and enable DMA autoincrement */
outport(PAR,0x1000);/* set DMA address */
for (i = 0; i < 10; i++)
    x = inport(PDR); /* read integer from memory */
```

Halting the DSP32 was not necessary, using an 0x19 code for the PCR would have kept the DSP32 running.

[1] Registed trademark of Borland.

Processor-Processor Serial Communication

The following program segments in DSP32 assembly language show how to send 32 data words of 32 bits from DSP32A to DSP32B via the serial port. DSP32B should be started first. The programs will loop at the wait labels until 32 words have been sent by processor A.

DSP32A Program:

```
main:   ioc = 0 /* disable serial port */
        dauc = 0
        pout = FirstAdr /* output serial port DMA pointer */
        ioc = 0x4FCF /* DMA using DSP32 clocks */
wait:   r1 = pout + (-LastAdr)
        if (mi) goto wait
        nop /* wait for 32 words */
```

DSP32B Program:

```
main:   ioc = 0 /* disable serial port */
        dauc = 0
        pin = FirstAdr /* input serial port DMA pointer */
        ioc = 0x2FCC /* DMA using DSP32 clocks */
wait:   r1 = pin + (-LastAdr)
        if (mi) goto wait
        nop /* wait for 32 words */
```

The nop instructions are included because the DSP32 always executes the instruction after a branch. This is due to the pipelining of the instructions.

REFERENCES

[1] J. Adams and M. Fishetti. Star wars sdi: the great experiment. *IEEE Spectrum*, 22(9):34–64, 1985.

[2] E. L. Althouse and M. A. Grimm. *Communication Networking in Support of the Strategic Defense Initiative.* NRL Memorandum Report XXX, Naval Research Laboratory, 1987.

[3] AT&T. *DSP32 Digital Signal Processor.* Allentown, PA, 1986.

[4] AT&T. *DSP32-SL Support Software Library.* Allentown, PA, 1987.

[5] H. A. Bethe, R. L. Garwin, K. Gottfried, and H. W. Kendall. Space-based ballistic-missile defense. *Scientific American*, 251(4):39–49, 1984.

[6] R. Bryant. Simulation on a distributed system. *First Inter. Conf. on Distributed Computing Systems*, 544–552, 1979.

[7] J. Cain, S. Adams, M. Noakes, P. Knoke, and E. Althouse. A distributed link assignment (reconstitution) algorithm for space-based sdi networks. In *Milcom'87*, pages 29.2.1–29.2.7, Washington, D.C., 1987.

[8] J. Cain, S. Adams, M. Noakes, T. Kryst, and E. Althouse. A near-optimum multiple path routing algorithm for space-based sdi networks. In *Milcom'87*, pages 29.3.1–29.3.7, Washington, D.C., 1987.

[9] A. B. Carter. The command and control of nuclear war. *Scientific American*, 252(1):32–39, 1985.

[10] Y. C. Chow, R. Newman-Wolfe, C. McLochlin, and C. Ward. Link assignment and routing strategies for rapidly changing satellite networks. In *SPIE's Symposium on Innovative Science and Technology*, Los Angles, CA, 1988.

[11] L. P. Clare, C. Y. Wang, and M. W. Atkinson. Multiple satellite networks: performance evaluation via simulation. In *Milcom'87*, pages 20.1.1–20.1.8, Washington, D.C., 1987.

[12] G. B. Dantzig. *Linear Programming and Extensions.* Princeton University Press, Princeton, NJ, 1963.

[13] V. Demjanenko and M. L. Craner. Simulation of a distributed communications network using a multi-tasking uniprocessor. In *Milcom'87*, pages 14.6.1–14.6.5. Washington, D.C., 1987.

[14] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[15] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *J. of the ACM*, 19(2):248–264, 1972.

[16] T. Evans and D. Smith. Optimally reliable graphs for both edge and vertex failures. *Networks*, 16:199–204, 1986.

[17] S. Even. *Graph Algorithms*. Computer Science Press, Inc., Rockville, MD, 1979.

[18] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, NJ, 1962.

[19] I. T. Frisch. An algorithm for vertex-pair connectivity. *Int. J. Control*, 6(6):579–593, 1967.

[20] J. Garcia-Luna-Aceves. A new approach to hierarchical routing in large networks. In *Milcom'87*, pages 10.7.1–10.7.7, Washington, D.C., 1987.

[21] A. Itai, Y. Perl, and Y. Shiloach. The complexity of finding maximum disjoint paths with length constraints. *Networks*, 12:277–286, 1982.

[22] D. B. Johnson. Efficient algorithms for shortest paths in sparse networks. *J. of the ACM*, 24(1):1–13, 1977.

[23] Y. Kajitani and S. Ueno. The minimum augmentation of a directed tree to a k-edge-connected directed graph. *Networks*, 16:181–197, 1986.

[24] B. Kaldenbach, D. R. Geissler, and E. W. Ver Hoef. A system simulator for low orbit satellite communication networks. In *Milcom'87*, pages 14.5.1–14.5.5, Washington, D.C., 1987.

[25] C. McLochlin, C. Ward, Y. C. Chow, R. Newman-Wolfe, and F. Gentges. An optimal link assignment and survivable routing strategy for large satellite networks. In *submitted Milcom'89*, 1989.

[26] C. McLochlin, C. Ward, Y. C. Chow, R. Newman-Wolfe, J. N. Wilson, and T. B. Hughes. Determining the delay and reliability of low altitude satellite network topologies using simulation. In *Symposium on the Simulation of Computer Networks*, pages 28–35, Colorado Springs, CO, 1987.

[27] C. McLochlin, C. Ward, Y. C. Chow, R. Newman-Wolfe, J. N. Wilson, and T. B. Hughes. Optimizing the delay and reliability of low altitude satellite network topologies. In *Milcom'87*, pages 20.3.1–20.3.6, Washington, D.C., 1987.

[28] C. McLochlin, C. Ward, Y. C. Chow, J. N. Wilson, and R. Newman-Wolfe. *Closed Forms and Algorithms for Determining Propagation Delay of Low Altitude Satellite Networks*. CIS Dept. TR 87-2, University of Florida, 1987.

[29] E. Minieka. *Optimization Algorithms for Networks and Graphs*. Marcel Dekker, Inc., New York City, NY, 1978.

[30] A. Moffat and T. Takaoka. An all pairs shortest path algorithm with expected time $o(n^2 \log n)$. *SIAM J. on Computing*, 16(6):1023–1031, 1987.

[31] J. Peacock, J. Wong, and E. Manning. Distributed simulation using a network of microcomputers. *Computer Networks*, 3-1:44–55, 1979.

[32] A. Polivka, E. Lawandales, H. Greene, and T. Blake. Survivable space-based c^2 communications network performance. In *Milcom '87*, pages 20.2.1–20.2.5, Washington, D.C., 1987.

[33] D. Ronen and Y. Perl. Heuristics for finding a maximum number of disjoint bounded paths. *Networks*, 14:531–544, 1984.

[34] H. Siegel and S. Smith. An interconnection network for multimicroprocessor emulator systems. *First Inter. Conf. on Distributed Computing Systems*, 772–782, 1979.

[35] J. W. Suurballe. Disjoint paths in a network. *Networks*, 4:125–145, 1974.

[36] J. W. Suurballe and R. E. Tarjan. A quick method for finding shortest pairs of disjoint paths. *Networks*, 14:325–336, 1984.

[37] Christopher Ward. *Link Assignment and Ground Coverage for Rapidly Changing Satellite Networks.* Dissertation, University of Florida, 1987.

[38] C. Weitzman. *Distributed Micro/Minicomputer Systems.* Printice-Hall, Inc., Englewood Cliffs, NJ, 1980.

[39] B. Ziegler. *Theory of Modelling and Simulation.* Wiley Interscience, New York, NY, 1976.

BIOGRAPHICAL SKETCH

Mr. McLochlin received the degrees of BSEE and MSEE from Purdue University, West Lafayette, Ind., in 1971 and 1972 respectively. In 1979 he received an advanced certificate of engineering from the Johns Hopkins Evening College, Baltimore, Md. He has worked for the Department of Defense for 14 years. He currently is employed as a consultant to the DoD.

While at the University of Florida, Mr. McLochlin has written several published papers in the areas of digital signal processing and low altitude satellite networks. He is president of *DSP Applications, Inc.*, a company specializing in high-speed digital signal processing applications.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Yuan-Chieh Chow, Chairman
Professor of
Computer and Information Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Richard Newman-Wolfe, Cochairman
Assistant Professor of
Computer and Information Sciences

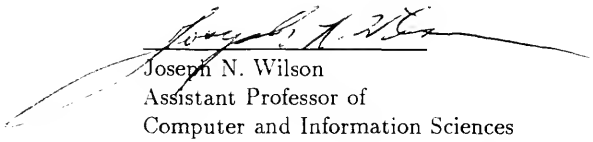
I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Jose C. Principe
Associate Professor of
Electrical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Paul Fishwick
Assistant Professor of
Computer and Information Sciences

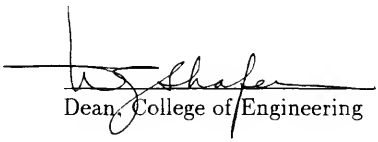
I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Joseph N. Wilson
Assistant Professor of
Computer and Information Sciences

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

May 1989



Dean, College of Engineering

Dean, Graduate School